Journal of Robotics and Control (JRC)
Volume 5, Issue 4, 2024
ISSN: 2715-5072, DOI: 10.18196/jrc.v5i4.22061

Visual Slam and Visual Odometry Based on RGB-D Images Using Deep Learning: A Survey

Van-Hung Le [1*]
[1] Information Technology Department, Tan Trao University, Vietnam
Email: [1] van-hung.le@mica.edu.vn
*Corresponding Author

***Abstract*—Visual simultaneous localization and mapping (Visual SLAM) based on RGB-D images includes two main tasks: building an environment map and simultaneously tracking the location/motion trajectory of the image sensor, or called visual odometry (VO). Visual SLAM and VO are used in many applications as robot systems, autonomous mobile robots, supporting systems for the blind, human-machine interaction, industry, etc. With the strong development of deep learning (DL), it has been applied and brought impressive results when building Visual SLAM and VO from image sensor data (RGB-D images). To get the overall picture of the development of DL applied to building Visual SLAM and VO systems. At the same time, the results, challenges, and advantages of DL models to solve Visual SLAM and VO problems. In this paper, we proposed the taxonomy to conduct a complete survey based on three methods from RGB-D images: (1) using DL for the modules (depth estimation, optical flow estimation, visual odometry, mapping, and loop closure detection) of the Visual SLAM and VO framework; (2) using DL modules to supplement (feature extraction, semantic segmentation, pose estimation, map construction, loop closure detection, others module) to Visual SLAM and VO framework; (3) using end-to-end DL to build Visual SLAM and VO systems. The studies were surveyed based on the order of methods, datasets, and evaluation measures, the detailed results according to datasets are also presented. In particular, the challenges of studies using DL to build Visual SLAM and VO systems are also analyzed and some of our further studies are also introduced.**

Keywords*—*Visual Slam; Visual Odometry(vo); Deep Learning (Dl); RGB-D Images; 3d Point Cloud Scene; Camera Pose; Trajectories Motion.

I. INTRODUCTION

Localization and mapping of the environment (3D space) for robots operating in the home [1], [2], [3] autonomous vehicles in factories [4], and blind people [5] are very important research in computer vision and robotics. These studies help entities locate themselves in the environment, understand the scene, and their navigation. To perform these tasks, it is necessary to solve two computer vision problems: Visual Simultaneous localization and mapping (Visual SLAM) and Visual Odometry (VO). Previously, the input data to build Visual SLAM and VO systems were Sound Navigation and Ranging (SONAR) sensors, 2D laser scanners, and Light Detection and Ranging (LiDAR) [5]. When using a LiDAR sensor, the results are accurate, but the cost of a LiDAR sensor is much more expensive than an image sensor [6]. In the 21st century, the development of computer hardware and image sensors has brought many newer and more affordable types of data such as monocular, stereo, or RGB-D. They can collect visual information about their surroundings. Therefore, the studies on visual SLAM and VO from RGB-D images are receiving very strong research attention.

Recently, with the advent of DL with some popular model types such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Generative Adversarial Networks (GAN), etc, which has brought impressive results in computer vision, DL has also been widely applied in research on visual SLAM and VO. At the same time, in recent years there have been many valuable surveys on visual SLAM [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23] and VO [24]. In which the input data for the studies surveyed above are all data obtained from image sensors, and DL is also the method with the best results in studies on Visual SLAM and VO.

A study of [14] presented a complete survey of Visual SLAM methods, in which the Visual SLAM construction model includes five steps (feature extraction, feature matching, pose estimation, loop closure, and map building) as shown in Fig. 1. In the study of Favorskaya et al. [23] is the most recent survey of Visual SLAM, in which the Visual SLAM process includes two main stages: VO and loop closure, when broken down in detail, it includes six steps: data pre-processing, feature extraction, feature matching, pose estimation, map building, and loop closure.

DL is also examined with three methods to implementing Visual SLAM: adding auxiliary modules based on DL, replacing modules with DL modules, and using end-to-end DL. However, most of the above surveys are based on statistics and classification of methods and datasets without examining in detail the algorithms and results of Visual SLAM and VO methods. At the same time, the advantages, disadvantages, and challenges of implementing Visual SLAM and VO have not been presented.

To have a detailed overview of the methods and results of Visual SLAM and VO methods, we have conducted a compre-

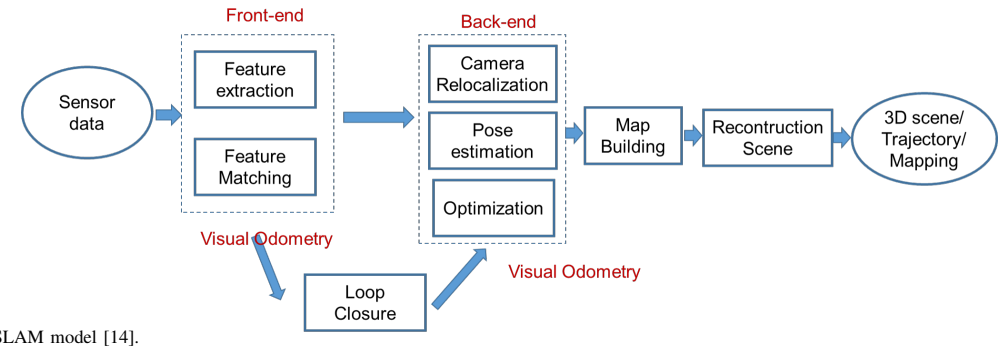

Fig. 1. General Visual SLAM model [14].

hensive and detailed survey of the methods, evaluation datasets, evaluation measures, results, advantages, and disadvantages, the challenges of the Visual SLAM and VO methods using DL with the input data from RGB-D images, as shown in Fig. 2.

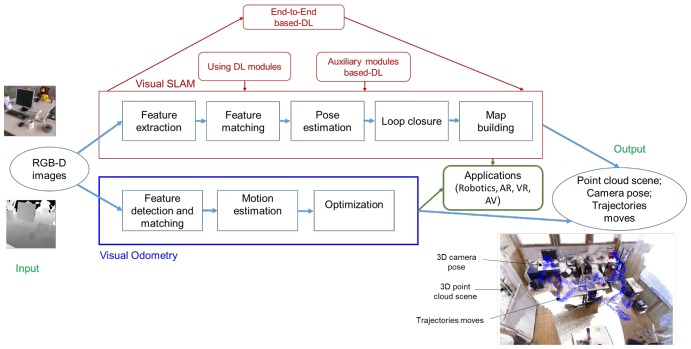

Fig. 2. The taxonomy of DL-based Visual SLAM and VO surveys from data obtained by the image sensors

Due to the data obtained the RGB-D image sensor provides data that is intuitive and close to the real environment surrounding the object. Especially in this study, we conduct a survey and analyze research in both the application direction of Visual SLAM and VO systems.

This paper includes the following main contributions:

- Proposing the taxonomy for investigating DL-based methods to perform Visual SLAM and VO methods from RGB-D image sensors.

- Conducting a complete survey based on three methods to construct Visual SLAM and VO systems from RGB-D images: (1) using DL modules to add auxiliary to the Visual SLAM and VO frameworks; (2) using DL modules to supplement the Visual SLAM and VO systems; (3) using end-to-end DL to build Visual SLAM and VO systems.

- The surveyed studies are detail examined and presented in the following order: methods, evaluation datasets, evaluation measures, results, discussions, and analyses.

- Presenting challenges in implementing DL-based Visual SLAM and VO with input data obtained from RGB-D sensors.

II. Related Work

Visual SLAM and VO surveys are not a new research problem. In the past 6 years, we found 18 valuable research papers on Visual SLAM and VO surveys, these studies are presented in Table I.

By the Visual SLAM categories, Mokssit et al. [22] and [21] have done a very valuable survey of DL techniques for Visual SLAM. In the research of Mokssit et al. [22], the authors proposed a taxonomy of four DL-based learning methods: modular learning, joint learning, confidence learning, and active learning. Modular learning includes learning depth to estimate the depth of the scene; learning optical flow is the process of determining optical flow (the process of determining the movement of a camera or object in the scene). The above three techniques all use two strategies of machine learning (ML): supervised learning and self-supervised learning. The fourth technique is learning to map, which is the process of building an environment map similar to the real world (3D reconstruction scene) and 3D object modeling with three types of maps: space-free maps, geometric maps, and semantic maps. The fifth technique is loop closure detection, which is the process of detecting loop closure frames. Joint learning is a learning method that can exploit the complete dependency relationship between different modules of the learning model to create a more accurate complete learning model. They are often based on two groups of techniques: only depth, optical flow, and ego-motion are grouped and optimized together; using end-to-end DL techniques. The third learning method is confidence learning, which is a learning method that can solve the uncertainty problem of DL with two techniques: uncertainty reduction and uncertainty estimation. The final learning method is activated learning. For a robot to work well in an environment that can find its way, understand the environment, and update localization and environmental maps, the robot's learning system must have two capabilities: active exploration to reduce the number of operations active when controlling robot operations and active perception to collect information from the environment and reduce sensor errors. The authors have listed a series of studies according to each

TABLE I. SURVEYS BY THE VISUAL SLAM AND VO FROM 2017- 2023(NOVEMBER).

Authors	Years	Methods	Type of datasets	Survey of DL
Taket et al. [7]	2017	Visual SLAM, VO	RGB-D	No
Jinyu et al. [8]	2019	Visual-inertial SLAM, VO	Stereo, RGB-D	No
Lai et al. [9]	2020	Visual SLAM, VO	RGB-D	Yes
Azzam et al. [10]	2020	Visual SLAM	RGB-D	Yes
Xia et al. [11]	2020	Semantic SLAM, Visual SLAM	Monocular, RGB-D, Stereo	Yes
Fang et al. [12]	2021	Visual SLAM	RGB-D	Yes
Barros et al. [13]	2022	Embedded SLAM, Visual-inertial SLAM, Visual-SLAM, VO	RGB-D	Yes
Abaspur et al. [14]	2022	Visual SLAM, VO	Sonar, Laser, LiDAR, RGB-D, Monocular, Stereo	Yes
Qin et al. [15]	2022	Visual SLAM	RGB-D	Yes
Zhang et al. [16]	2022	Visual SLAM	RGB-D	Yes
Tsintotas et al. [17]	2022	Visual SLAM, VO	RGB-D	Yes
Chen et al. [18]	2022	Semantic Visual SLAM	Sonar, Laser, LiDAR, RGB-D, Monocular, Stereo	Yes
Tian et al. [19]	2022	Visual SLAM, VO	RGB-D, GPS	No
Tourani et al. [20]	2022	Visual SLAM	RGB-D	Yes
Agost et al. [24]	2022	VO	LiDAR, RGB-D, Point cloud	Yes
Dai et al. [21]	2023	Visual SLAM, VO	RGB-D	Yes
Mokssit et al. [22]	2023	Visual SLAM	Monocular, RGB-D, Stereo	Yes
Favors et al. [23]	2023	Visual SLAM, VO	Monocular, RGB-D, Stereo	Yes
Our	2023	Visual SLAM, VO	RGB-D	Yes

learning technique and method, the studies are surveyed based on the methods, goals, data architectures, advantages, and disadvantages. Barros et al. [13] conducted a survey on Visual SLAM algorithms, including three methods based on output data: visual-only SLAM, Visual-inertial SLAM, and RGB-D SLAM. For each method, a timeline is presented. Finally, datasets for evaluating Visual SLAM algorithms are presented. In more detail, research by Chen et al. [18] surveyed semantic Visual SLAM that meets the requirements of accuracy and real-time. The authors investigated three methods: object detection, semantic segmentation, and instance segmentation to extract semantic information from the environment. Jinyu et al. [8] conducted a survey and evaluated the algorithms of Visual-inertial SLAM. The basic theories of Visual SLAM and Visual-inertial SLAM are presented. The most important content is filtering-based methods and optimization-based methods presented to solve the problem of building Visual SLAM and Visual-inertial SLAM systems. Finally, the KITTI [25], EuRoC [26], TUM VI [27], ADVIO [28], and VICON [8] datasets are listed and used to evaluate constructed Visual-inertial SLAM models. Tourani et al. [20] presented a survey based on 45 recent outstanding studies of Visual SLAM in which recent advancements and impressive results. The results of Visual SLAM are analyzed and discussed based on the novelty domain, objectives, employing algorithms, and semantic level. At the same time, the existing challenges and trends of Visual SLAM systems are also presented. Favorskaya et al. [23] presented the state-of-the-art Visual SLAM systems, in which the Visual SLAM system construction model was also surveyed and presented with a very detailed method by the DL techniques. At the same

time, prominent datasets for evaluating Visual SLAM models are also listed and briefly described. By only VO categories, Agostinho et al. [24] have conducted a complete and detailed survey of VO systems used for robots and autonomous vehicles operating indoors. The authors presented this state-of-the-art VO framework according to models, algorithms, and results. The results show an increase in accuracy of 33.14% for building trajectory from point cloud data. At the same time, challenges when building a VO system are also discussed and presented.

By the application of Visual SLAM and VO categories, Theodorou et al. [29] surveyed the applications of Visual SLAM for localization, mapping, and wayfinding, in which applications are presented according to Visual SLAM algorithms with three methods: monocular-based (based on the image sequence), stereo-based (based on the camera trajectory and building a map of the environment based on feature points), combination of monocular-based and stereo-based (based on image sequences or feature points for mapping, tracking, and wayfinding). Research by Lai et al. [9] surveyed methods for building Visual SLAM and VO systems according to two methods: traditional and DL.

III. Visual SLAM and VO Using Deep Learning: Survey

As shown in Fig. 2, the paper surveys Visual SLAM and VO based on the RGB-D images captured from image sensors are introduced. In this study, we only surveyed studies conducted based on DL.

A. Deep Learning-based module for Visual SLAM and Visual Odometry

In the study of Mokssit et al. [22], the Visual SLAM framework includes the following modules: depth estimation, optical flow, VO, mapping, and loop closure detection. In study of Favorskaya et al. [23] presents the survey according to the architecture of DL. In this paper, we present the modules for the architecture of DL to build the Visual SLAM system. We present a survey on DL usage with the following modules.

1) Depth estimation

a. Methods

By the depth estimation module, David et al. [30] proposed a deep network consisting of 2 stacks to directly regress depth: using the coarse-scale network to estimate the global structure of the scene, using the fine-scale network to refine it using local information. Chen et al. [31] proposed a deep network which is a variation of Hourglass to estimate depth by training a multi-scale deep network and relative depth annotations of the data. The input for pixel-wise depth prediction is a single image. Zhou et al. [32] proposed an end-to-end learning deep network for a single-view depth (scene structure) and pose estimation (camera motion) from the image sequence. To predict single-view depth, the DispNet network architecture was used in an encoder-decoder design. To predict the camera pose, the target view is concatenated with all the source views according to the color channel of the input image sequence. Wang et al. [33] proposed a method to improve the method performance of Zhou et al. [32] for estimating depth and camera pose by the CNN-based with a simple normalization step, thereby significantly improving the performance of depth estimation. In the proposed method, the authors applied a Direct Visual Odometry (DVO) [34] pose predictor to predict the output pose based on the input dense depth map, thereby reducing information loss of sequence frames during scene reconstruction. Garg et al. [35] proposed an unsupervised CNN learning method based on auto-encoder architecture to predict single-view depth without requiring learning from annotated ground-truth depths. The loss function of CNN is to represent the difference between the source image and the inverse warped target image show the correlation of prediction error and align two different depths without using ground-truth of depth maps. Godard et al. [36] proposed an end-to-end unsupervised DL to estimate monocular depth with a new information of loss function that enforces left-right depth consistency inside the network. The information loss function is capable of combining three error information: smoothness/disparity smoothness loss, reconstruction/appearance matching loss, and left-right disparity consistency terms/left-right disparity consistency loss. The loss function is the sum of the above three error information. Casser et al. [37] proposed an unsupervised DL method based on exploiting 3D geometry structure and semantics to build a model for estimating scene depth and ego-motion. The input of the learning method is a sequence of RGB frames and performs the following calculation steps: object masks, object ego-motion, and individual object motion. The output of the learning model is the image warped according to ego-motion. Bian et al. [38] proposed an unsupervised DL for estimating depth and motion from two consecutive frames of monocular video. The feature used for the training process is a geometry consistency constraint extracted from a self-discovered mask of dynamic scenes and occlusions to enforce scale consistency. From which the motion of a global scale can be estimated.

b. Datasets

KITTI Dataset: The KITTI dataset [39], [40], [41] is the most popular dataset for evaluating Visual SLAM and VO models and algorithms. This dataset includes two versions: the KITTI 2012 dataset [40] and the KITTI 2015 dataset [41]. Geiger et al. [39] data are used to evaluate the VO, object detection, and object tracking models. KITTI 2012 dataset is collected from two high-resolution camera systems, a Velodyne HDL-64E laser scanner (grayscale and color), and a state-of-the-art OXTS RT 3003 localization system (a combination of devices such as GPS, GLONASS, security IMU, and RTK

correction signals). This database is also divided into data sets serving different problems. The data set used to evaluate the optical flow estimation model includes 194 image pairs for training and 195 image pairs for testing, the images have a resolution of 1240×376 pixels, and the ground-truth data is built based on 50% dense. The dataset used to evaluate the 3D visual odometry / SLAM model consists of 22 stereo sequences collected from a length of 39.2 km of driving. This data set provides benchmarks and evaluation measures for VO and Visual SLAM such as motion trajectory and driving speed. The dataset used to evaluate object detection and 3D orientation estimation, the ground-truth data includes accurate 3D bounding boxes for classifying object classes such as 'Cars', 'Vans', 'Trucks', 'Pedestrians', 'Cyclists', and 'Trams'. The ground-truth data of 3D objects in point cloud data were manually labeled to evaluate algorithms for 3D orientation estimation and 3D tracking. Geiger et al. [40] provide a raw dataset for evaluating stereo, optical flow, and object detection models. The data collection system is built based on the following devices: camera images, laser scans, high-precision GPS measurements, and IMU accelerations. The data collection context is very diverse, the system captures real-world traffic situations and ranges from highways through rural areas to inner-city scenes with many static objects and dynamic. This dataset includes color and grayscale image data, saved as 8bit ".png". The second type of data is OXTS (GPS/IMU), with each frame 30 different values are stored as text files of the following information: the geographic coordinates including altitude, global orientation, velocities, accelerations, angular rates, accuracies, and satellite information. The third data type is the Velodyne scans, which are stored as floating point binaries. This dataset also provides ground-truth data including 3D bounding box trackless annotation with represented in Velodyne coordinates and labels of object classes are 'Car', 'Van', 'Truck', 'Pedestrian', 'Person (sitting)', 'Cyclist', 'Tram' and ' Misc'. Menze et al. [41] published the KITTI 2015 dataset for evaluating the optical flow algorithms. The annotation data provided the annotation data of the 3d object in the scene, where the construction of the annotation data is based on the process of recovering the static elements of the scene and inserting them into the moving object using the 3D CAD model into the scene by Google 3D Warehouse.

NYUDepth dataset [42]: This dataset consists of 1449 RGB-D images collected from MS Kinect from multiple buildings in three US cities. They include 464 different indoor scenes belonging to 26 scene classes. The dataset contains 35,064 distinct objects, spread across 894 different classes. For each of the 1449 images, supporting captions were added manually.

Make3D dataset [43]: This dataset is collected with 534 pair images (RGB images and depth maps), the resolution of the RGB images is 2272×1704 and the size of the depth map is 55×305. Training data includes 400 images, testing data includes 134 images collected from a 3d scanner. In addition, 588 images were also collected from the Internet. Algorithm evaluation data is based on a person not part of the project collecting data of the environment with images larger than 800×600 of scenes at 'Campus', 'Garden', 'Park', 'House', 'Building', 'College', 'University', 'Church', 'Castle', 'Court', 'Square', 'Lake', 'Temple', and 'Scene'.

Cityscapes dataset [44]: To evaluate object detection and classification models, especially when using DL models with outdoor environments. Ramos et al. [44] have published the Cityscapes dataset. The data were collected from stereo cameras using 1/3 in CMOS 2 MP sensors (OnSemi AR0331) in 50 different cities. The original data consists of 5000 manually annotated images from 27 cities for dense pixel-level. In addition, there are 20,000 raw pixel-level annotated images for evaluating object detection using object boundaries.

TUM RGB-D SLAM dataset [45]: The authors collected the TUM RGB-D SLAM dataset using the MS Xbox Kinect sensor, the collected data consists of RGB-D frame sequences. The environment for data collection includes two different indoor scenes. The first is a typical office environment called "fr1" with a size of $6 \times 6\ m^2$, and the second is a large industrial hall called "fr2" with a size of $10 \times 12\ m^2$. The ground-truth trajectory from the motion capture system is provided by eight high-speed tracking cameras. This dataset includes 39 frame sequences and divided into four groups: "Calibration", "Testing and Debugging" (fr1/xyz, fr1/rpy, fr2/xyz, fr2/rpy), "Hand-held SLAM" (fr1/360, fr1/floor, fr1/desk, fr1/desk2, fr1/room, fr2/360 hemisphere, fr2/360 kidnap, fr2/desk, fr2/desk with person, fr2/large no loop, fr2/large with loop), and "Robot SLAM" (fr2/pioneer 360, fr2/pioneer slam, fr2/pioneer slam2, fr2/pioneer slam3). The RGB-D images are 640×480 pixels in size and captured at a rate of 30Hz.

ICL-NUIM dataset [46]: This is a dataset consisting of RGB-D sequences used to evaluate VO, 3D reconstruction, and SLAM algorithms collected from the living room and the office room. For each scene, the authors collect four frame sequences: living room (kt0, kt1, kt2, kt3), office room (kt0, kt1, kt2, kt3), the number of frames in each sequence is different. The ground-truth camera trajectories (POVRay) and synthetic trajectories data are obtained from the ground-truth depth maps and color images. In this dataset, there are two main types of noise: noise from RGB images and noise from depth images when collecting data with MS Kinect.

c. Evaluation Measure

To evaluate depth estimation models, the RMSE(Root Mean Squared Error) measure is often used. RMSE is the square root of the average of the squared errors and is the standard deviation of the residuals (prediction error). The residual is a measure of distance from the regression line data points; RMSE is a measure of how spread out these residuals are the other words,

it tells you how concentrated the data is around the line of best fit. RMSE includes $RMSE_{linear}$ expressed in formula (1), and $RMSE_{log}$ expressed in formula (2).

$$RMSE_{linear} = \sqrt{\frac{|| \sum_{i=1}^{N} y_i - y_i^* ||^2}{N}} \quad (1)$$

$$RMSE_{log} = \sqrt{\frac{|| \sum_{i=1}^{N} logy_i \quad - \quad logy_i^* ||^2}{N}} \quad (2)$$

where N is the number of data points, y_i is the predicted depth map, and y_i^* is the ground-truth of the depth map. $RMSE_{linear}$ and $RMSE_{log}$ have values as small as possible.

d. Results and Discussions

The results of evaluating depth estimation on the KITTI [39], NYUDepth [42], Make3D [43], Cityscapes [44], TUM RGB-D SLAM [45], and ICL-NUIM [46] datasets with $RMSE_{linear}$, $RMSE_{log}$ measurements are shown in Table II. The evaluation results of DRM-SLAM_F [47] on the NYUDepth dataset [42] are the best ($RMSE_{linear} = 0.42, RMSE_{log} = 0.16$). The evaluation results of Cowan-GGR [48] on the KITTI dataset [39] are the best ($RMSE_{linear} = 3.923, RMSE_{log} = 0.188$). The evaluation results of DVO_CNN [33] on the Make3D dataset [43] are the best ($RMSE_{linear} = 3.923, RMSE_{log} = 0.188$). The evaluation results of DRM-SLAM_F [47] on the TUM RGB-D SLAM [45] dataset are the best ($RMSE_{linear} = 0.62, RMSE_{log} = 0.23$). The evaluation results of PE_N [49] on the ICL-NUIM [46] dataset are the best ($RMSE_{linear} = 0.22, RMSE_{log} = 0.12$). In Table II, the KITTI dataset is evaluated in most studies, and the Make3D and Cityscapes datasets are evaluated in only a few studies. Table II also shows studies on depth estimation evaluated across multiple datasets, so equal comparisons across studies are difficult to make. Therefore, Table II has many empty cells.

2) Optical Flow Estimation

a. Methods for Optical Flow Estimation

By the optical flow estimation module, Dosovitskiy et al. [67] proposed and compared two end-to-end CNN architectures for optical flow estimation from a pair of images: FlowNetSimple and FlowNetCorr. They are called FlowNet. FlowNetSimple uses a generic network with two stacked input images to extract motion information for optical flow prediction. The FlowNetCorr creates two identical streams for each image of the input image pair and then combines the two streams to predict the optical flow. Ilg et al. [68] proposed a deep network to improve the FlowNet of Dosovitskiy et al. [67] for optical flow estimation, it is called FlowNet 2.0. The proposed method includes three important improvements: first is concerned with the training data, it is trained on the FlyingChairs dataset and FlyingThings3D dataset to exploit the quality of training data for optical flow estimation. The second is to develop a stacked architecture to warp with the previously estimated flow. The third is to address small displacements by introducing a sub-network specializing in small motions. This improved version makes the accuracy increase four times and the speed increases more than 17 times. Ranjan et al. [69] proposed a method by applying the spatial-pyramid formula to DL, with the idea of applying a coarse-to-fine method to calculate and update the flow at each pyramid level by warping an image of a pair image. The number of parameters of this network is reduced by 96% compared to FlowNet by applying the Spatial Pyramid Network and the flow at each pyramid level applies a convolutional network to pairs of warped images and the learned convolution filters are applied like spatial-temporal filters into the network to improve the FlowNet network. Sun et al. [70] proposed PWC-Net which is a combination of pyramidal processing, warping, and a cost volume for optical flow estimation. It is an improved model from Spatial Pyramid Network [69] and FlowNet 2.0 [68]. The input of PWC-Net is still an image pair, the CNN features of the second image are calculated based on the current optical flow of the first image. The warped features of the image pair are used to construct a cost volume. PWCNet's calculation time is only 1/17 of FlowNet 2.0. Teed et al. [71] proposed the RAFT network for optical flow estimation. RAFT includes (1) the per-pixel features of image pairs extracted using a feature encoder module, 4D correlation volume is built and synthesized from all pairs of feature vectors using a feature encoder module, (3) update module is iterated on recurrently optical flow by lookups on the correlation volumes. Ren et al. [72] proposed an unsupervised DL network called Dense Spatial Transform Flow (DSTFlow) that estimates optical flow based on input frame pairs. This is an end-to-end learning consisting of three components: localization layer, sampling layer, and interpolation layer. Backpropagation is used to train the parameters in all three layers. Zhu et al. [73] proposed an unsupervised CNN framework to estimate optical flow based on proxy ground-truth data. This data is responsible for guided optical flow learning and consists of two stages: the first is a ground-truth flow proxy created based on classical approaches, and the second is the process of fine-tuning the model using image-minimizing reconstruction loss. Wang et al. [74] proposed an end-to-end deep neural network to estimate optical flow based on learning large motions using occlusion models clearly and a new warping. The main flow of this method is to use two copies of FlowNetS to share parameters and estimate forward and backward optical flow. Janai et al. [75] proposed a new unsupervised learning framework for optical flow estimation based on multiple frames by exploiting the temporal relationship between frames and occlusions jointly. The flow fields and occlusion map are estimated based on evaluating the loss function of the warped images. Zhong et al. [76] proposed an unsupervised learning network for optical flow estimation, called Deep Epipolar Flow. It uses soft epipolar constraints on the low level and subspace of the scene when not

TABLE II. DEPTH ESTIMATION RESULTS BASED ON DL.

Authors/Years	Dataset/ Measu./ Methods	NYUDepth [42]		KITTI 2012 [39]		Make3D [43]		Cityscapes [44]		TUM RGB-D SLAM dataset [45]		ICL-NUIM dataset [46]	
		RMSE _linear	RMSE _log	RMSE _linear	RMSE _log	RMSE _linear	RMSE _log	RMSE _linear	RMSE _log	RMSE _linear	RMSE _log	RMSE _linear	RMSE _log
David et al. [30]/2014	Multi-Scale DN	2.19	0.285	5.246	0.248	8.325	0.409	-	-	-	-	-	-
Liu et al. [50]/2015	CRF_ CNN	0.82	-	-	-	-	-	-	-	-	-	-	-
Zoran et al. [51]/2015	Ordinal Relationships DN	1.2	0.42	-	-	-	-	-	-	-	-	-	-
Wang et al. [52]/2015	HCRF_CNN	0.75	0.26	-	-	-	-	-	-	-	-	-	-
Eigen et al. [53]/2015	SGD_DN	0.64	0.23	-	-	-	-	-	-	1.41	0.37	0.83	0.43
Liu et al. [54]/2016	CRF_CNN_N	0.73	0.33	-	-	-	-	-	-	0.86	0.29	0.81	0.41
Chen et al. [31]/2016	pixel-wise_ ranking DN	0.24	0.38	-	-	-	-	-	-	-	-	-	-
Laina et al. [55]/2016	Deeper FCRN	0.51	0.22	-	-	-	-	-	-	1.07	0.39	0.54	0.28
Garg et al. [35]/2016	Unsupervised CNN	-	-	5.104	0.273	9.635	0.444	-	-	-	-	-	-
Godard et al. [36]/2017	Unsupervised CNN_D	-	-	6.125	0.217	8.86	0.142	14.445	0.542	-	-	-	-
Zhou et al. [32]/2017	SfMLearner	-	-	4.975	0.258	10.47	0.478	-	-	-	-	-	-
Weer. et al. [49]/2017	PE_S	0.52	0.21	-	-	-	-	-	-	0.69	0.25	0.32	0.18
Weer. et al. [49]/2017	PE_N	0.45	0.17	-	-	-	-	-	-	0.65	0.24	0.22	0.12
Mal et al. [56]/2018	StD	0.48	0.17	-	-	-	-	-	-	0.7	0.27	0.36	0.18
Chen et al. [57]/2018	RSS	0.45	0.18	-	-	-	-	-	-	0.65	0.24	0.33	0.19
Yang et al. [58]/2018	pre-trained KITTI + Cityscapes	-	-	6.641	0.248	-	-	-	-	-	-	-	-
Wang et al. [33]/2018	DVO_CNN	-	-	5.583	0.228	8.09	0.204	-	-	-	-	-	-
Yang et al. [58]/2018	pre-trained KITTI	-	-	6.5	0.27	-	-	-	-	-	-	-	-
Mahj. et al. [59]/2018	pre-trained KITTI	-	-	6.22	0.25	-	-	-	-	-	-	-	-
Yin et al. [60]/2018	Geonet-VGG pre-trained KITTI	-	-	6.09	0.247	-	-	-	-	-	-	-	-
Yin et al. [60]/2018	Geonet-Resnet pre-trained KITTI	-	-	5.857	0.233	-	-	-	-	-	-	-	-
Zou et al. [61]/2018	DF-Net pre-trained KITTI	-	-	5.507	0.223	-	-	-	-	-	-	-	-
Mahj. et al. [59]/2018	pre-trained KITTI + Cityscapes	-	-	5.912	0.243	-	-	-	-	-	-	-	-
Yin et al. [60]/2018	Geonet-Resnet pre-trained KITTI + Cityscapes	-	-	5.737	0.232	-	-	-	-	-	-	-	-
Zou et al. [61]/2018	DF-Net pre-trained KITTI + Cityscapes	-	-	5.215	0.213	-	-	-	-	-	-	-	-
Mal et al. [56]/2018	StD- RGB	0.51	0.21	-	-	-	-	-	-	-	-	-	-
Chen et al. [57]/2018	RSS-RGB	0.73	0.19	-	-	-	-	-	-	-	-	-	-
Ranjan et al. [62]/2019	pre-trained KITTI + Cityscapes	-	-	5.199	0.213	-	-	-	-	-	-	-	-
Ranjan et al. [62]/2019	Pre-trained KITTI	-	-	5.326	0.217	-	-	-	-	-	-	-	-
Bian et al. [38]/2019	pre-trained KITTI	-	-	5.439	0.217	-	-	-	-	-	-	-	-
Bian et al. [38]/2019	pre-trained KITTI + Cityscapes	-	-	5.234	0.208	-	-	-	-	-	-	-	-
Casser et al. [37]/2019	struct2depth	-	-	5.291	0.215	-	-	-	-	-	-	-	-
Godard et al. [63]/2019	Monodepth2	-	-	4.701	0.19	-	-	-	-	-	-	-	-
Ye et al. [47]/2020	DRM-SLAM_F	0.42	0.16	-	-	-	-	-	-	0.62	0.23	0.3	0.13
Rares et al. [64]/2020	packnet-sfm	-	-	4.601	0.189	-	-	-	-	-	-	-	-
Ye et al. [47]/2020	DRM-SLAM_C	0.5	0.19	-	-	-	-	-	-	0.7	0.28	0.36	0.18
Luo et al. [65]/2020	EPC++	-	-	5.35	0.216	-	-	-	-	-	-	-	-
Lee et al. [66]/2021	Faster R-CNN AVN	-	-	4.772	0.191	-	-	-	-	-	-	-	-
Mumuni et al. [48]/2022	Cowan-GGR	-	-	3.923	0.188	-	-	-	-	-	-	-	-
Mumuni et al. [48]/2022	Cowan	-	-	4.916	0.212	-	-	-	-	-	-	-	-

in motion. The unsupervised training process is optimized based on image-based losses and epipolar constraint losses. Liao et al. [77] proposed a method to estimate optical flow based on a combination of utilizing intrinsic image decomposition and recomposition based on Retinex theory on two consecutive frames of outdoor UAV videos and an edge refinement scheme based on weighted neighborhood filtering. Yan et al. [78] proposed a semi-supervised DL network to estimate optical flow. The proposed network is based on direct estimation from real data without using ground-truth data. Foggy images and optical flow modules are estimated from clean images based on domain transformation. These two data sources interact with each other, the optical flow module and the flow map that produces the flow map must be the same to generate the same error. Dai et al. [79] proposed a self-supervised learning framework for depth and object motion estimation. In which the motion of individual objects is predicted based on rotation and translation of 6 DOF. The proposed network consists of two subnets: ObjMotion-net and the Depth-net. The pose network is used to design ObjMotion-net and Depth-net are designed based on the encoder and the decoder structure with the basic structure being ResNet50. Ranjan [62] proposed an unsupervised training framework of multiple specialized neural networks called Competitive Collaboration to perform depth estimation, camera motion estimation, optical flow, and segmentation. This general framework solves the problem by dividing the scene into moving objects and static background, camera motion, depth of static scene structure, and optical flow of moving objects.

b. Datasets of Optical Flow Estimation

MPI Sintel dataset [80]: To evaluate optical flow estimation models, Butler et al. [80] have published the MPI Sintel dataset. This dataset is created from 3D animations built from Sintel open-source code. Based on the cartoon, the camera's parameters, moving objects, and graphics are all calculated using vectors. The ground-truth data for optical flow estimation is also provided in the form of vectors. This dataset includes 35 clips, with 23 clips (1064 frames) used for the training model and 12 clips (564 frames) used for the testing model. The process of dataset creation is carried out in three different ways: first is "*Albedo*", this data uses the simplest pass of constant color with almost no lighting effect; the second is "*Clean*", this data using this pass adds complexity by introducing various types of lighting that make smooth gloss surfaces, self-shadowing, darkening in cavities and darkening where the object is close to the surface; the third is "*Final*", this data is similar to the released film and adds some effects such as atmospheric effects, depth of field blur, motion blur, and color correction.

Middlebury dataset [81]: Unlike other datasets, this dataset has a very small number of frames, consisting of only 8 frames, and the ground-truth data is determined in the middle pair. The authors not only collected color images but also created grayscale images. The data is divided into 12 sequences for the training model with ground-truth data, and 12 sequences for the testing model.

Flying Chairs dataset [67]: The ground-truth data is the model of the chair. This data includes 22,872 image pairs and corresponding flow fields. Among them, 964 images were collected from Flick with the environments 'City', 'Landscape', and 'Mountain' with a resolution of 1024×768. From this ground-truth image, the authors cropped the images with the dimensions are 512×384 in 4 quadrants. Chair objects are added to the background, resulting in 809 chair types with 62 views per chair.

Foggy dataset [78]: This is a synthetic dataset built by combining the defogging method with the original FlowNet2 [68], PWCNet [70], and CC [62]. The defogging method was proposed by Berman et al. [82]. The generated data includes 2,346 real fog image pairs used for training and the ground-truth includes 100 real fog image pairs that are annotated manually.

c. Evaluation Measure of Optical Flow Estimation

To evaluate the results of the optical flow estimation, the methods often use the EPE (End-Point Error) losses measure between the predicted optical flow and ground-truth. The unit of measurement is pixels. Based on the evaluation measure, if the EPE is small, the optical flow estimation model is better.

d. Results, Discussions of Optical Flow Estimation

The results of optical flow estimation are shown in Table III. The results were evaluated on seven datasets when evaluated on the Sintel Clean dataset [80], the best results were with the method of Liao et al. [77] (FlowNet2-IAER). On the Sintel Final dataset [80], the best results are from the method of Liao et al. [77] (FlowNet2-IAER). On the KITTI 2012 dataset [39], the best result is that of the method of Zhong et al. [76] (sub-test-ft). On the KITTI 2015 dataset [41], the best results are from the method of Ren et al. [72]. On the Middlebury dataset [81], the best results are from the method of Bailer et al. [83]. On the Flying Chairs dataset [67], the best result is that of the method of Zhu et al. [73]. Finally, on the Foggy dataset [78], the best result is that of the method of Yan et al. [78]. The results show that more recent studies tend to have lower error rates. However, studies often focus on evaluating a few datasets: Sintel Clean, Sintel Final, KITTI 2012, and KITTI 2015, so there are many empty results on the remaining datasets.

3) Keypoints Detection and Feature Matching

a. Methods for Keypoints Detection and Feature Matching

By the keypoints detection and feature matching categories, Hart et al. [84] proposed a method to detect keypoints and feature matching by finding a description prediction model before performing matching. In which the points are well located and repeatable, which also reduces the number of points of interest and reduces the time to consider points for the

TABLE III. THE OPTICAL FLOW ESTIMATION RESULTS BASED ON DL

Datasets/Authors/Years	Sintel Clean [80]	Sintel Final [80]	KITTI 2012 [39]	KITTI 2015 [41]	Middlebury [81]	Flying Chairs [67]	Foggy [78]
	Train/Test (EPE)	Train/Test (EPE)	Train/Test (EPE)	Train/Test (EPE)	Train/Test (EPE)	Train/Test (EPE)	Train/Test (EPE)
Dosovitskiy et al. [67]/2015	3.20/6.08	4.83/ 7.88	6.07/7.6	-	3.81/4.52	-	-
Ilg et al. [68]/2017	1.45/ 4.16	2.01/ 5.74	1.28/ 1.8	2.30/-	0.35/0.52	-	-
Ranjan et al. [69]/2017	3.17/6.64	4.32 /8.36	8.25/10.1		0.33/ 0.58	-/3.07	-
Ren et al. [72]/2017	4.17/5.30	5.45/6.16	3.29 / 4.0	0.36/0.39	-	-	-
Zhu et al. [73]/2017	-/3.01	-/7.96	-/9.5	-	-	-/3.01	-
Sun et al. [70]/2018	2.02/4.39	2.08/5.04	1.45/1.7	2.16/-	-	-	-
Wang et al. [74]/2018	4.03/7.95	5.95/9.15	3.55/4.2	8.88/-	-	-/3.76	-
Janai et al. [75] (Hard) /2018	5.38/8.35	6.01/9.38	-	8.8/-	-	-	-
Janai et al. [75] (Hard-ft) /2018	6.05/-	7.09/-	-	7.45/-	-	-	-
Janai et al. [75] (None-ft) /2018	4.74/-	5.84/-	-	3.24/-	-	-	-
Janai et al. [75] (Soft-ft) /2018	3.89/7.23	5.52/8.81	-	3.22/-	-	-	-
Zhong et al. [76] (baseline) /2019	6.72/-	7.31/-	3.23/-	4.21/-	-	-	-
Zhong et al. [76] (gtF) /2019	6.15/-	6.71/-	2.61/-	2.89/-	-	-	-
Zhong et al. [76] (F) /2019	6.21/-	6.73/-	2.56/-	3.09/-	-	-	-
Zhong et al. [76] (low-rank) /2019	6.39/-	6.96/-	2.63/-	3.03/-	-	-	-
Zhong et al. [76] (sub) /2019	6.15/-	6.83/-	2.62/-	2.98/-	-	-	-
Zhong et al. [76] (sub-test-ft) /2019	3.94/6.84	5.08/8.33	2.61/1.1	2.56/-	-	-	-
Zhong et al. [76] (sub-train-ft) /2019	3.54/7.0	4.99/8.51	2.51/1.3	2.46/-	-	-	-
Bailer et al. [83] /2019	-/3.748	-/5.81	-/3.5	-	-/0.33	-/2.45	-
Yan et al. [78] /2020	-	-	-/1.6	-	-	-	-/4.32
Liao et al. [77] (PWC-Net-ft) /2021	2.02/4.39	2.08/5.04	1.45/1.7	-	-	-	-/6.10
Liao et al. [77] (FlowNet2-ft) /2021	1.45/4.16	2.01/5.74	1.28/1.8	-	-	-	-/4.74
Liao et al. [77] (FlowNet2-IA) /2021	1.52/4.11	5.51/1.4	1.4/1.8	-	-	-	-/4.72
Liao et al. [77] (FlowNet2-IAER) /2021	1.46/4.06	2.13/1.37	1.37/1.8	-	-	-	-/5.19

matching process. Verdie et al. [85] proposed a method that allows detecting keypoints and feature matching based on a training method to identify potentially stable points on the training image by creating a regression set of points of a score map whose values are local maxima at these locations. Shen et al. [86] proposed an end-to-end matching network based on improvements by LF-Net, the proposed method proposes a scale space structure with the corresponding map for keypoints detection. Second, the training patch is selected based on the general loss function and neighbor mask.

b. Datasets for Keypoints Detection and Feature Matching Evaluation

To evaluate the results of keypoints detection and feature matching, Verdie et al. [85] used the following datasets.

Webcam dataset [85]: This is a dataset consisting of 6

scenes of which five scenes (StLouis, Mexico, Chamonix, Courbevoie, and Frankfurt) are selected from the AMOS [87] dataset and the Panorama scene is collected from the roof with a 360 degrees view.

Oxford dataset [88]: This is a small dataset, consisting of 8 scenes (viewpoint changes (1) and (2); scale changes (3) and (4); image blur (5) and (6); JPEG compression (7); and illumination (8)). On the data, there are two types of changes: scene type and image condition. In a scene, there are two types of variable regions: (a) containing uniform regions with distinctive edge boundaries and (b) the other containing repeating motifs in different forms.

EF dataset [89]: This is a small dataset, consisting of 5 sequences of 38 images which contains drastic illumination and background clutter changes.

HPatches(HP) dataset [90]: This is a dataset consisting of 116 sequences built from 6 images with known patterns in nature and man-made, it is divided into two parts: (1) HP-viewpoint includes 59 sequences with significant viewpoint changes; (2) HP-illumination includes 57 sequences with significant illumination changes. The data of these two sets is divided into 90% for the training and validation model and 10% for the testing model. During the training process, the data is standardized to a size of 320×240.

c. Evaluation Measure of Keypoints Detection and Feature Matching

To evaluate the results of the keypoints detection and feature matching process, the studies often use the measure of repeatability, of which there are two evaluation cases: the first is to evaluate the repeatability in the case of taking the ratio. The main score is 2% of the score on the image (*2%*); the second is that a keypoint can not be used more than once when evaluating repeatability (*stand.*). Based on the results of these two measurements, the higher the result is the better [85]. Another measure, average match score (AMS), is also evaluated for keypoints detection and feature matching [90].

d. Results of Keypoints Detection and Feature Matching

The keypoints detection and feature matching results based on DL are shown in Table IV. In addition, the authors also compared some traditional methods such as Fast [91], SFOP [92], SIFER [93], SIFT [94], SURF [95], WADE [96], and EdgeFoci [97]. When comparing the measures *(2%)* and *(stand.)*, TILDE [85] has the best results when compared to measure *AMS*, and RF-Net has the best results. However, the results are evaluated on multiple datasets and with different measures, so many cells in Table IV are empty.

4) DL Module Adds to the Visual Slam Framework

a. Feature Extraction Module DL

Qin et al. [101] proposed a keypoint extraction network used to resemble the ORB-SLAM2 [102] module in the VO framework. Therefore, SP-Flow is used to replace ORB-SLAM2 in the VO framework. This network is called SP-Flow, it is a combination of a self-supervised framework and the Lucas–Kanade. The self-supervised framework of SP-Flow includes three stages: keypoint pre-training, keypoint self-labeling, and joint training. In the Visual SLAM model, whether feature extraction is effective or not often depends on the feature point extraction for a single image and its feature point matching accuracy between two successive frames. SP-Flow has tried to make the feature extraction process simple but still ensure accuracy. The architecture of SP-Flow includes six conventional convolution layers. Bruno et al. [103] proposed a module the Learned Invariant Feature Transform (LIFT) in the traditional ORB-SLAM of the Visual SLAM method. This module is responsible for extracting features from images in the ORB-SLAM [104] method. The architecture of LIFT is based on CNN consisting of three modules: detector, orientation estimator, and descriptor. It is a very important module in the ORB-SLAM method, and LIFT has been pre-trained on many VO datasets.

Studies based on this method have performed evaluations on datasets such as the TUM RGB-D SLAM dataset [45], the KITTI 2012 dataset [39], and the Euroc dataset [105]. The TUM RGB-D SLAM [45], and the KITTI 2012 [39] datasets have been presented above.

The Euroc dataset [105] is collected on-board a Micro Aerial Vehicle (MAV), its data includes stereo images, synchronized IMU measurements, accurate motion, and ground-truth structure. This dataset is used to evaluate the Visual-inertial SLAM and 3D reconstruction capabilities. The data includes 11 stereo sequences collected from slow flights under good visual conditions to dynamic flights with motion blur and poor illumination. This dataset includes two data types: images collected from industrial scenarios; and images collected from inside a Vicon motion capture system, with obstacles placed over the scene.

To evaluate the results of the Visual SLAM algorithm using the DL module for feature extraction, the methods use several evaluation metrics as follows: (1) The absolute trajectory error (ATE) [45] is the distance error between the ground-truth $\hat{AT}_i$ and the estimated motion AT_i trajectory, aligned with an optimal $SE(3)$ pose $\mathbf{T}$. ATE is calculated according to formula (3).

$$ATE = \min_{T \in SE(3)} \sqrt{\frac{1}{I_{gt}} \sum_{i \in I_{gt}} ||TAT_i - \hat{AT}_i||^2} \quad (3)$$

(2%) t_{rel} and r_{rel} measurements: t_{rel} is the average transnational $RMSE$ drift (%) on length of 100m-800m. r_{rel} is the average rotational RMSE drift (°/100m) on length of 100m-800m.

Results of the Visual SLAM method when using the feature

TABLE IV. THE KEYPOINTS DETECTIONS AND FEATURE MATCHING RESULT BASED ON DL

Authors/Years	Datasets/ Measu./ Methods	Webcam	Oxford		EF			HP-viewpoint	HP-illumination
		2%	stand.	2%	stand.	2%	Average match score	Average match score	Average match score
Low et al. [94]/2004	SIFT	20.7	46.5	43.6	32.2	23	0.296	0.49	0.494
Rosten et al. [91]/2006	Fast	26.4	53.8	47.9	39	28	-	-	-
Bay et al. [95]/2006	SURF	29.9	56.9	57.6	43.6	28.7	0.235	0.493	0.481
Forstner et al. [92]/2009	SFOP	22.9	51.3	39.3	42.2	21.2	-	-	-
Zitnick et al. [97]/2011	EdgeFoci	30	54.9	47.5	46.2	31	-	-	-
Mainali et al. [93]/2013	SIFER	25.7	45.1	40.1	27.4	17.6	-	-	-
Salti et al. [96]/2013	WADE	27.5	44.3	51	25.6	28.6	-	-	-
Verdie et al. [85]/2015	TILDE-GB	33.3	54.5	32.8	43.1	16.2	-	-	-
Verdie et al. [85]/2015	TILDE-CNN	36.8	51.8	49.3	43.2	27.6	-	-	-
Verdie et al. [85]/2015	TILDE-P24	40.7	58.7	59.1	46.3	33	-	-	-
Verdie et al. [85]/2015	TILDE-P	48.3	58.1	55.9	45.1	31.6	-	-	-
Tian et al. [98]/2017	L2-Net+DoG	-	-	-	-	-	0.189	0.403	0.394
Tian et al. [98]/2017	L2-Net+SURF	-	-	-	-	-	0.307	0.627	0.629
Tian et al. [98]/2017	L2-Net+FAST	-	-	-	-	-	0.229	0.571	0.431
Tian et al. [98]/2017	L2-Net+ORB	-	-	-	-	-	0.298	0.705	0.673
Tian et al. [98]/2017	L2-Net+Zhang et al.	-	-	-	-	-	0.235	0.685	0.425
Mishchuk et al. [99]/2017	Hard-Net+DoG	-	-	-	-	-	0.206	0.436	0.468
Mishchuk et al. [99]/2017	Hard-Net+SURF	-	-	-	-	-	0.334	0.65	0.668
Mishchuk et al. [99]/2017	Hard-Net+FAST	-	-	-	-	-	0.29	0.617	0.63
Mishchuk et al. [99]/2017	Hard-Net+ORB	-	-	-	-	-	0.238	0.616	0.632
Mishchuk et al. [99]/2017	Hard-Net+Zhang et al.	-	-	-	-	-	0.273	0.671	0.557
Ono et al. [100]/2018	LF-Net	-	-	-	-	-	0.251	0.617	0.566
Shen et al. [86]/2019	RF-Net	-	-	-	-	-	0.453	0.783	0.808

extraction module using DL are shown in Table V. The results are based on the ATE, t_{rel}, and r_{rel} measurements that the smaller the better. The results are evaluated on three datasets with three types of measures, each method only evaluates one dataset and one type of measure. Therefore, Table V still has many empty results. Table V also shows that the error results on the TUM RGB-D SLAM and Euroc datasets are very low (0.03-0.5m), but the results on the KITTI 2012 dataset are very large (8-11m). This proves that choosing a standard dataset for evaluating the feature extraction problem also has many challenges.

b. DL Semantic Segmentation Module

Sun et al. [107] proposed MR-SLAM to improve the results of the RGB-D SLAM. The main idea of this method is to use the RGB-D data-based motion removal method and integrate it into the front end of the RGB-D SLAM framework. The input data of MR-SLAM is RGB-D images, the ego-motion compensated image difference is first used to detect moving objects, a particle filter is second used to detect motion, and third is to apply the maximum-a-posterior (MAP) estimator on vector quantified depth images to construct the foreground. Kaneko et al. [108] proposed a framework to improve the efficiency of the Visual SLAM framework by using the results of mask-based semantic segmentation to identify feature point extraction regions (detect and segment several objects on the image). The object mask problem is implemented using DeepLab v2 [109]. This helps to reduce the number of incorrect matches between correspondences when using RANSAC. The authors applied to ORB-SLAM framework to build a Visual SLAM system. Yu et al. [110] proposed DS-SLAM to improve localization efficiency in dynamic environments when performing pose estimation. DS-SLAM has five threads running in parallel: tracking, semantic segmentation, local mapping, loop closing, and dense semantic map creation. In particular, the local mapping thread and loop closing thread are implemented similarly to the ORB-SLAM2 [102] method. DS-SLAM uses the raw RGB image is utilized for semantic segmentation and moving consistency checks simultaneously by SegNet and RANSAC, respectively. Finally, The global octo-tree map is built based on the combination of the created local point clouds from the keyframes' transformation matrix and the depth images. Bescos et al. [111] proposed DynaSLAM based on ORB-SLAM2 method. DynaSLAM's input data is in dynamic scenarios for monocular, stereo, and RGB-D images. DynaSLAM can detect moving objects using multi-view geometry, DL, or both types of models. The pixel-wise semantic segmentation of dynamic objects with stereo and monocular input data is performed using Mask R-CNN, with RGB-D data using the multi-view geometry method for rendering. The mapping and tracking steps are performed based on the ORB-SLAM2 method. Zhong et al. [112] proposed Detect-SLAM based on integrating ORB-SLAM2, an

TABLE V. RESULTS OF VISUAL SLAM METHOD WHEN USING THE DL FEATURE EXTRACTION MODULE

Authors/Years	Dataset/ Measurements/ Methods	TUM RGB-D SLAM dataset [45]	KITTI 2012 dataset [39]			Euroc dataset [105]
		Absolute Trajectory Error (ATE)(m)	t_rel(%)	r_rel(deg/100m -^{o})	Absolute Trajectory Error (ATE)(m)	Absolute Trajectory Error (ATE)(m)
Mur et al. [102]/2017	ORB-SLAM2 (stereo)	-	0.727	0.22	-	-
Tang et al. [106]/2019	GCN-SLAM	0.05	-	-	-	-
Qin et al. [101]/2020	SP-Flow SLAM	0.03	-	-	-	-
Qin et al. [101]/2020	Stereo LSD-SLAM	-	0.942	0.272	-	-
Qin et al. [101]/2020	SP-Flow SLAM(stereo)	-	0.76	0.19	-	-
Bruno et al. [103]/2020	LIFT-SLAM	-	-	-	9.19	0.573
Bruno et al. [103]/2021	LIFT-SLAM (fine-tune KITI)	-	-	-	11.33	0.08
Bruno et al. [103]/2021	LIFT-SLAM (fine-tune Euroc)	-	-	-	8.94	0.07
Bruno et al. [103]/2021	Adaptive LIFT-SLAM	-	-	-	8.56	0.04
Bruno et al. [103]/2021	Adaptive LIFT-SLAM (fine-tune KITI)	-	-	-	11.24	0.28
Bruno et al. [103]/2021	Adaptive LIFT-SLAM (fine-tune Euroc)	-	-	-	11.3	0.048

object detection module using a Single Shot Multibox Object Detector (SSD) to perform. Detect-SLAM also includes three parallel streams: tracking, local mapping, and loop closing. However, there are the following new points. Firstly, Detect-SLAM only cares about moving objects. The second is that the static objects are reconstructed on keyframes according to the point cloud data and the object map is also constructed. The third is to improve object detection results using a SLAM-enhanced detector. Tian et al. [113] proposed a novel framework to build the Visual SLAM system based on the combination of Faster RCNN for object detection, semantic segmentation in 3D space, and the estimation results from the SLAM system. The input data of the framework is an RGB-D image. Firstly, the local target map is built using CNN to detect the 2D object proposals. Then, the dynamic global target map is updated based on the local target map obtained by CNNs. Finally, the detection result of the current frame is obtained by projecting the global target map into 2D space. Cheng et al. [114] proposed the OFB-SLAM method to improve the results of the Visual SLAM system in the case of a dynamic environment. OFB-SLAM uses optical flow in a feature-based monocular SLAM system to remove dynamic feature points on the input frame. OFB-SLAM includes two modules: ego-motion estimation and dynamic feature point detection. Ego-motion estimation module extracts the feature points from the current frame and the previous frame, to find the corresponding feature pair between the two frames, RANSAC is used. Optical flow is used to detect object motion. OFB-SLAM is integrated into ORB-SLAM and implements the next steps of the Visual SLAM system. Shao et al. [115] proposed a method to filter outliers of RANSAC-based F-matrix calculations using faster R-CNN. In which the inliers are trained using semantic patches tailored which can provide semantic labels of image regions. From there, low-quality feature areas are effectively reduced. The proposed method is added to the ORB-SLAM system. Xua et al. [116] proposed Deep SAFT to improve feature-based vSLAM's applicability in more challenging environmental conditions. Deep SAFT is an online learning scene adaptation feature transform that is capable of self-adapting to recently observed scenes by taking advantage of the advantages of CNN. The authors used Deep SAFT to replace ORB-SLAM2 in the Visual SLAM system. Liu et al. [117] proposed the Edge-Feature Razor (EF-Razor) method, EF-Razor first uses semantic information offered by the real-time object detection method YOLOv3 to distinguish edge features. To effectively filter unstable features onto the SLAM system, EF-Razor was used. The authors integrated EF-Razor into ORB-SLAM2. Rusli et al. [118] proposed a semantic SLAM using method objects and walls as a model of an environment, called RoomSLAM. RoomSLAM includes two modules running in parallel: front-end and back-end. The front-end performs object detection and wall detection by using YOLOv3 on RGB images. The depth images are converted to point cloud data to determine the location of objects and walls in the 3D space/the real world. They are seen as landmarks of the environment and the walls are used to construct rooms in the scene. The back-end is responsible for estimating the state through the optimization graph. In RoomSLAM a second component that is also very important in the room, RoomSLAM also looks for similarities between rooms to detect loop closures. Jin et al. [119] proposed an unsupervised semantic segmentation

SLAM framework, called USS-SLAM to improve Robot positioning accuracy when moving, this framework is integrated into the ORB-SLAM2 method. To do this, USS-SLAM filters out dynamic features using a semantic segmentation model learned from the DeepLab V2 whose backbone is ResNet. This learning method can be trained by the adversarial transfer learning method in multi-level feature spaces. The next steps of the Visual SLAM system are based on ORB-SLAM2. Zhao et al. [120] proposed a semantic Visual-inertial SLAM system for dynamic environments based on VINS-Mono [121] with three streams: RGB-image manager, semantic segmentation manager, and feature point processing. In particular, RGB-image manager and semantic segmentation manager use the RGB images and the semantic segmentation result. The feature point processing flow uses the optical flow to track feature points on the RGB images. The output of this research is that it is possible to perform real-time trajectory estimation by utilizing the pixel-wise results of semantic segmentation. Cheng et al. [122] proposed DM-SLAM based on feature-based methods to improve the results of the location accuracy in dynamic environments. DM-SLAM is combined from an instance segmentation network with optical flow information. DM-SLAM includes four modules: semantic segmentation, ego-motion estimation, dynamic point detection, and a feature-based SLAM framework. The semantic segmentation module uses Mask R-CNN for object segment segmentation, followed by moving points being detected and removed in ego-motion estimation. The dynamic feature points are extracted from the dynamic point region detected in the previous step. Finally, the feature-based SLAM framework module uses ORB-SLAM2. Liu et al. [123] proposed RDS-SLAM to improve the results of building Visual SLAM systems in real-world dynamic environments. RDS-SLAM proposes a semantic segmentation thread that does not have to wait for results from any module, and the tracking thread also does not have to wait for results from the segmentation module. This method helps to effectively perform semantic segmentation results for dynamic object detection and eliminate outliers. The next implementation of RDS-SLAM is based on ORB-SLAM3. Su et al. [124] proposed a real-time Visual SLAM algorithm based on deep learning based on the ORB-SLAM2 method. To extract semantic information from the images, a parallel semantic thread is built. To remove dynamic features in the image, the authors used an optimized optical flow mask module. Dynamic objects in images are detected using YOLOv5s built into the semantic thread. To improve the system results in the tracking module, a method of optimizing the homograph matrix is used.

To evaluate the DL module for semantic segmentation added to the Visual SLAM system, the studies evaluated the following datasets. CARLA [125] is used to study the results of three methods of autonomous driving: a classic modular pipeline, an end-to-end model trained via imitation learning, and an end-to-end model trained via reinforcement learning. CARLA can provide automated digital environments such as urban layouts, buildings, and vehicles. This can support the development, training, and validation of urban automated driving systems.

ObjectFusion dataset I, ObjectFusion dataset II, ObjectFusion dataset III, and ObjectFusion dataset VI [113] are collected from the Asus Xtion Pro RGB-D sensor in indoor environments. Trajectories are chosen to build data with many prominent objects as keyframes both locally and globally. ObjectFusion dataset I is that there is an object in each frame of the scene, the data is a frame sequence consisting of 1,801 frames. ObjectFusion dataset II is that there are multiple objects in each frame of the scene such as 'Chairs', 'Dogs', 'Pot plants', and so on. The data is collected in full lighting conditions and is a frame sequence consisting of 1,625 frames. ObjectFusion dataset III is collected in a more challenging context, the data is collected in a scene with many objects and is occluded in many frames. ObjectFusion dataset VI is similar to ObjectFusion dataset III, and the data is captured in a scene with many objects and moving obstacles.

The ICL-NUIM dataset [126] is an RGB-D benchmarking used to evaluate system VO and Visual SLAM algorithms. The image data was compiled from camera trajectories in raytraced 3D models in POVRay with two scenes in the living room and office. They provided some ground-truth data. Living room ground-truth data includes 3D surface ground-truth together with the depth maps, camera poses, and camera trajectory. In addition, the ground-truth data of 3D reconstruction is also provided to evaluate.

ADVIO dataset [28] is collected from an iPhone, a Google Pixel Android phone, and a Google Tango device in different indoor and outdoor scenes. It includes 23 sequence frames (7 sequences collected from Office indoor scenes, 12 sequences collected from urban indoor scenes, 2 sequences collected from urban outdoor scenes, and 2 sequences collected from suburban outdoor scenes). The ground-truth data includes ground-truth trajectory based on the camera pose calculated from the IMU data of the iPhone.

To evaluate the DL module for semantic segmentation added to the Visual SLAM system, the studies evaluated the following measurements. Mean Tracking Rate (MTR) [108] is the tracking success rate in 50 trials when successful tracking is performed on 80% of the 1000 frames of the sequence, as computed in the formula (4).

$$MTR = \frac{1}{m} \sum_{i=1}^{m=50} (TrackingRate_i) \tag{4}$$

where $TrackingRate_i$ is the "Tracking Rate" (%) at time i^{th}, and m is the number of times the "Tracking Rate" is performed.

Mean Trajectory Error (MTE) [108] is an estimate of the camera's position relative to the defined ground-truth. It is

the error distance for each time step and the average value of a sequence as "Trajectory Error (m)". Here, only calculate MTE for "Success Tracking" i.e. "Tracking Rate" exceeds 80%, MTE is computed based on the formula (5).

$$MTE = \frac{1}{m}\sum_{i=1}^{m}(\frac{1}{n_i}||X_t - Y_{it}||_2) \quad (5)$$

where $i = 1, 2, ..., 50$ is the number of "Successful Tracking" trials. X_t is the 3D position of the ground-truth trajectory, and Y_{it} is the 3D position of the estimated trajectory over the entire time series $(t = 1, ..., n_i)$. ATE is presented in Eq. (3).

IOU (Intersection over Union) is a measure to evaluate the results of detecting objects in the scene during the process of building the Visual SLAM system, IOU is calculated according to the formula (6).

$$IOU = \frac{BB_g \cap BB_r}{BB_g \cup BB_r} \quad (6)$$

where BB_g is the ground-truth bounding box of the object, and BB_r is the bounding box prediction of the object. Pixel accuracy (PA) is calculated according to the formula (7).

$$PA = \frac{\sum_i n_{ii}}{\sum_i t_i} \quad (7)$$

Mean precision (MP) is calculated according to the formula (8).

$$MP = \frac{1}{n_{cl}}\frac{\sum_i n_{ii}}{\sum_j n_{ji}} \quad (8)$$

where n_{ij} is the number of pixels classified as j while the true value is i. n_{cl} is the total classes. t_i is the number of pixels that belong to class i with $t_i = \sum_j n_{ij}$.

Another measure used is the absolute translation error RMSE (Tabs) [116], which is the distance between the estimated trajectory and the ground-truth trajectory. Another type of measure used is the absolute translation (trans.) error RMSE (Tabs) [116], which is the distance between the estimated trajectory and the ground-truth trajectory. The calculation of RMSE (Tabs) was performed in the study of Sturm et al. [45].

In Zhao et al. [120] study, measurements such as RMSE, Mean error, and Absolute Pose Error(APE) are also presented. These measures are also defined [127].

Results of Visual SLAM when using the DL semantic segmentation module are presented in Table VI. The results have been evaluated on multiple measures (MTR, MTE, ATE, IOU, PA, MP, RMSE (Tabs), RMSE, Mean Error, APE) with multiple datasets (CARLA [125], TUM RGB-D SLAM dataset [45], ObjectFusion dataset I, ObjectFusion dataset II, ObjectFusion dataset III, ObjectFusion dataset VI [113], ICL-NUIM dataset [126], ADVIO dataset [28]). Although, they all use DL for semantic segmentation and are added to the Visual SLAM system, each dataset and method uses a different measure. Therefore, there are many empty results in Table VI.

c. DL Pose Estimation Module

Pose estimation is the process of estimating the camera pose as the subject carrying the camera moves in the environment/scene. In this section, we survey methods and research on using DL for camera pose estimation. Zou et al. [128] proposed an ObjectFusion system to estimate the camera pose of each RGB-D frame and build 3D object surface reconstruction in the scene. To do this, the instance segmentation masks are detected in each frame and used to encode each object instance to a latent vector by a deep implicit object representation. To detect each object instance, the object shape and pose are initialized. The camera pose is estimated based on the deep implicit object representation and sparsely sampled map points. Xu et al. [129] proposed MID-Fusion for a multi-instance dynamic RGBD SLAM system. The authors used an object-level octree-based volumetric representation to estimate the camera pose in a dynamic environment. Mumuni et al. [48] proposed a confidence-weighted adaptive network (Cowan) framework to train a depth estimation model from monocular RGB images and predict camera pose, and optical flow by using EgoMNet, and OFNet, respectively. Cowan's training process includes two stages: the first is DepthNet, EgoMNet, and OFNet to predict the outputs depth map, camera pose, and optical flow, respectively. The second is that the outputs from the previous step are used to filter suitable regions allowing the network to be updated again in the previous step. Zhu et al. [130] proposed a method to learn neural camera pose representation coupled with neural camera movement representation in the 3D scene. The camera pose is represented by a vector, the local camera movement is represented by a matrix operating on the vector of the camera pose. The vector representing the camera pose includes 6 DOFs, with information such as position and direction of movement. The regression camera pose output is through the DL network. Qiao et al. [131] proposed Objects Matter for camera re-localization in the scene, the proposed method is based on extracting object relation features and strengthening the inner representation of an image using an Object Relation Graph (ORG). Where the objects in the image and the relationships between them can be important information to restore the camera pose. To extract features of objects, the proposed method uses Graph Neural Networks (GNNs) and then integrates ORG into PoseNet and MapNet to be able to predict many datasets. To evaluate the DL module for pose estimation/camera pose estimation added to the Visual SLAM system, the studies evaluated the following datasets. SceneNet RGB-D [132] is a large synthetic dataset with a 5M indoor synthetics video dataset of high-quality ray-traced RGB-D images, built-in full lighting conditions, and providing ground-truth data (3D ground-truth trajectories). The authors built a ground-truth trajectory with a length of 5 minutes for one journey, the image resolution is 320×240 pixels, resulting in 300 images in a trajectory. SceneNet RGB-D is used to evaluate

TABLE VI. RESULTS OF VISUAL SLAM WHEN USING THE DL SEMANTIC SEGMENTATION MODULE

Authors/Years	Methods/ Datasets/ Matrix	CARLA [125]	TUM RGB-D SLAM dataset [45]	Object Fusion -dataset I [113]	Object Fusion -dataset II [113]	Object Fusion -dataset III [113]	Object Fusion -dataset IV [113]	ICL-NUIM dataset [126]	ADVIO dataset [28]		
		MTR (%)/ MTE (m)	ATE	Mean IOU / Mean PA/ MP	Mean IOU / Mean PA/ MP	Mean IOU / Mean PA/ MP	Mean IOU / Mean PA/ MP	Absolute trans. error RMSE (Tabs)	RMSE	Mean Error	APE for trans.
Sun et al. [107]/2017	MR-SLAM	-	0.085	-	-	-	-	-	-	-	-
Kaneko et al. [108]/2018	Mask-SLAM	58.2/ 13.7	-	-	-	-	-	-	-	-	-
Yu et al. [110]/2018	DS-SLAM	-	0.103	-	-	-	-	-	-	-	-
Qin et al. [121]/2018	VINS-Mono	-	-	-	-	-	-	-	5.037	4.71	1.68
Bescos et al. [111]/2018	DynaSLAM	-	0.019	-	-	-	-	-	-	-	-
Zhong et al. [112]/2018	Detect-SLAM	-	0.113	-	-	-	-	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-FCN-VOC8s	-	-	0.52/ 0.62/ 0.729	0.5169/ 0.5966/ 0.7103	0.5775/ 0.6559/ 0.6708	0.3529/ 0.4168/ 0.7361	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-CRF-RNN	-	-	0.59/ 0.63/ 0.938	0.4769/ 0.4899/ 0.5633	0.5618/ 0.6058/ 0.4115	0.273/ 0.2989/ 0.5955	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-Mask-RCNN	-	-	0.59/ 0.64/ 0.895	0.4855/ 0.5021/ 0.7125	0.4946/ 0.5397/ 0.4489	0.3433/ 0.3938/ 0.716	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-Deeplabv3+	-	-	0.58/ 0.63/ 0.856	0.4849/ 0.4927/ 0.719	0.4869/ 0.537/ 0.4458	0.3484/ 0.3952/ 0.7351	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-SORS (GLOBAL)	-	-	0.71/ 0.726/ 0.954	0.5889/ 0.6438/ 0.7989	0.6063/ 0.6764/ 0.872	0.4012/ 0.4261/ 0.7806	-	-	-	-
Tian et al. [113]/2019	ObjectFusion-SORS (ACTIVATE)	-	-	0.702/ 0.724/ 0.936	0.5301/ 0.5765/ 0.8626	0.5528/ 0.6106/ 0.902	0.3728/ 0.3878/ 0.7873	-	-	-	-
Cheng et al. [114]/2019	OFB-SLAM	-	0.082	-	-	-	-	-	-	-	-
Shao et al. [115]/2020	Semantic Filter_ RANSAC_ Faster R-CNN	-	0.19	-	-	-	-	-	-	-	-
Xua et al. [116]/2020	Offline Deep SAFT	-	0.0179	-	-	-	-	0.057	-	-	-
Xua et al. [116]/2020	Continuous Deep SAFT	-	0.168	-	-	-	-	0.043	-	-	-
Xua et al. [116]/2020	Discrete Deep SAFT	-	0.0235	-	-	-	-	0.065	-	-	-
LIU et al. [117]/2020	EF-Razor	-	0.0168	-	-	-	-	-	-	-	-
Rusli et al. [118]/2020	RoomSLAM	-	0.205	-	-	-	-	-	-	-	-
Jin et al. [119]/2020	USS-SLAM with ALT	-	0.01702	-	-	-	-	-	-	-	-
Jin et al. [119]/2020	USS-SLAM without ALT	-	0.019	-	-	-	-	-	-	-	-
Zhao et al. [120]/2020	Visual-inertial _SS	-	-	-	-	-	-	-	4.84	4.51	1.61
Cheng et al. [122]/2020	DM-SLAM	-	0.034	-	-	-	-	-	-	-	-
Liu et al. [123]/2021	RDS-SLAM	-	0.065	-	-	-	-	-	-	-	-
Su et al. [124]/2022	ORB-SLAM2 _PST	-	0.019	-	-	-	-	-	-	-	-

semantic segmentation, instance segmentation, object detection, optical flow, camera pose estimation, and 3D scene labeling algorithms in the Visual SLAM system. 7-Scenes dataset [133] is collected from the handheld MS Kinect RGB-D sensor with a resolution of 640×480 pixels. The ground-truth data of the tracking camera and dense 3D model are built from the KinectFusion system based on the scene coordinate regression forest.

To evaluate the DL module for pose estimation added to the Visual SLAM system, the studies performed evaluation on the following measurements. ATE is defined in the formula (3). Dense Correspondence Re-Projection Error ($DCRE$) [131] is the 2D displacement magnitude according to the 2D projection of dense 3D points rendered by 3D ground-truth camera poses and predicted camera poses. Based on two measures ATE and $DCRE$, the smaller the value, the better the proposed method.

Results of Visual SLAM when using the DL pose estimation module are presented in Table VII. Just like the results above, in Table VII, the results of multiple methods are evaluated on three different datasets, so there are many empty result cells. The

number of evaluation methods on the SceneNet RGB-D dataset [132] is the largest, the results show a huge difference (with the ObjectFusion_S3 [128] method the result is 0.79, but the Maskfusion(MF)_S3 [134] results are 14.824) shown in Table VII.

d. DL Map Construction Module

Zhao et al. [139] proposed a deep network to build 3D dense mapping, called LKN-VO (Learning Kalman Network-based monocular VO). The input data of LKN-VO is monocular RGB images. The dense optical flow is estimated by FlowNet2, and the depth map is estimated by DepthNet. The global pose trajectory is built upon transferring and filtering 6 DOF relative poses using the $SE(3)$ composition layer. Next, the point cloud data of the image is built based on the depth map and the learned global pose. The output is that a dense 3D map is constructed.

Tao et al. [140] proposed a method for constructing an indoor 3D semantic Visual SLAM algorithm based on the combination of Mask Regional Convolutional Neural Network (RCNN) and ORB feature extraction. To accurately collect keypoints, the authors used the real-time ORB feature extraction. To detect instance segmentation tasks and semantic association of map points, the proposed method uses the Mask RCNN. The output is the constructed semantic map.

By the mapping categories, to build a safe path that can avoid obstacles in the environment for robots or autonomous vehicles, geometric maps need to be built based on spatial maps. This includes information about the space of the environment, and structures to plan movements, paths, and its location in the environment. Han et al. [141] surveyed building an environmental map. The process of building semantic mapping includes three modules: spatial mapping, acquisition of semantic information, and map representation.

McCormac et al. [142] proposed a method combining CNNs and Visual SLAM (ElasticFusion) to build dense 3D maps. In which Visual SLAM builds a 3D global map based on 2D images, CNNs perform semantic predictions on multiple views based on probability. The output is a densely annotated semantic 3D map.

Sunderhauf et al. [143] propose a method for building an environment map based on Visual SLAM and DL techniques for object detection and segmentation, thereby creating a semantic map of the environment with full geometry information of the environment and information on 3D objects in the environment. In the phase of detecting and segmenting objects in the environment, the method has been used and evaluated on many typical models such as Fast R-CNN, Faster R-CNN, YOLO, or the Single Shot MultiBox Detector (SSD).

Yang et al. [144] proposed a real-time semantic mapping system that includes two main tasks: the first is 3D geometric reconstruction often using SLAM models, and the second is 3D object semantic segmentation using a CNN model to convert pixel label distributions of 2D images to 3D grids and propose a Conditional Random Field (CRF) model with higher order cliques to enforce semantic consistency among grids.

Grinvald et al. [145] proposed online volumetric instance aware semantic mapping from RGB-D images based on geometric segmentation with object-like convex 3D segmentation of depth image using geometry-based method, semantic instance aware segmentation refinement is performed on the RGB image by Mask R-CNN. The data association is performed on RGB-D image pairs, and map integration is performed by Voxblox TSDF-based dense mapping framework.

Karkus et al. [146] suggested the Differentiable Mapping Network (DMN) based on a combination of spatial structure and an end-to-end training model for mapping. DMN performs the construction of maps that allow embedding views in a spatial structure. The filters are used to localize image sequences using particle filters. The gradient descent is used to combine the map representation and localization.

To evaluate the performance of the map construction DL module in the Visual SLAM system, the researchers used several datasets with RGB-D images as shown below. KITTI 2012 dataset [39], NYU RGB-D V2 dataset [42], TUM RGB-D SLAM dataset [45], and ICL-NUIM dataset [46] have been presented above.

Mask-RCNN MC dataset [140] is a self-generated dataset, including 10,000 images collected from 21 types of objects commonly found in homes and laboratories ('Person', 'Robot', 'Suitcase', 'Chair', 'Air conditioner', 'Desk', 'Bookcase', 'Cat', 'Jackboard', 'Door', 'TV', 'Potted plant', 'Book', 'Mouse', 'Dog', 'Umbrella', 'Drone', 'Bed', 'Laptop', 'Cell phone', 'Keyboard'). The data is collected based on an MS Kinect V2 connected to a Laptop with Intel i5-7500, the memory is 32 GB, and a GPU GTX 1080 graphic. They are mounted on a moving robot. The data is divided into 90% for training and 10% for testing.

To evaluate the performance of the map construction DL module in the Visual SLAM system, the researchers used several datasets with RGB-D images as shown below.

Measures t_{rel}, r_{rel}, and RMSE have been presented above. The average log error (log)(ALE) measure is calculated based on the formula (9).

$$ALE = \sqrt{\frac{1}{N}\sum_{p}(log(d_p^{gt} - log(d_p))^2} \tag{9}$$

The absolute relative error ($AbRE$) measure is calculated based on the formula (10).

$$AbRE = \frac{1}{N}\sum_{p}\frac{|d_p^{gt} - d_p|}{d_p^{gt}} \tag{10}$$

TABLE VII. RESULTS OF VISUAL SLAM WHEN USING THE DL POSE ESTIMATION MODULE

Authors/Years	Datasets/ Measurements/ Methods	SceneNet RGB-D [132]	KITTI 2012 dataset [39]	7-Scenes Dataset [133]
		RMSE of Absolute Trajectory Error (ATE) (cm)	RMSE of Absolute Trajectory Error (ATE) (cm)	Dense Correspondence Re-Projection Error (DCRE)(cm)
Kahler et al. [135]/2015	InfiniTAM(IM)_S1	22.486	-	-
Kahler et al. [135]/2015	InfiniTAM(IM)_S2	28.08	-	-
Kahler et al. [135]/2015	InfiniTAM(IM)_S3	13.824	-	-
Kahler et al. [135]/2015	InfiniTAM(IM)_S4	34.846	-	-
Dai et al. [136]/2017	BundleFusion (BF)_S3	4.164	-	-
Dai et al. [136]/2017	BundleFusion (BF)_S1	5.2	-	-
Dai et al. [136]/2017	BundleFusion (BF)_S2	5.598	-	-
Dai et al. [136]/2017	BundleFusion (BF)_S4	7.742	-	-
Kendall et al. [137]/2017	PoseNet17	-	-	24
Runz et al. [134]/2018	Maskfusion(MF)_S4	18.972	-	-
Runz et al. [134]/2018	Maskfusion(MF)_S1	20.856	-	-
Runz et al. [134]/2018	Maskfusion(MF)_S2	22.71	-	-
Brahmbhatt et al. [138]/2018	PoseNet + log q	-	-	22
Runz et al. [134]/2018	Maskfusion(MF)_S3	14.824	-	-
Brahmbhatt et al. [138]/2018	MapNet	-	-	21
Mahjourian et al. [59]/2018	Vid2Depth	-	1.25	-
Xu et al. [129]/2019	MID-fusion(MID)_S1	5.98	-	-
Xu et al. [129]/2019	MID-fusion(MID)_S2	4.132	-	-
Xu et al. [129]/2019	MID-fusion(MID)_S3	5.1675	-	-
Xu et al. [129]/2019	MID-fusion(MID)_S4	5.3825	-	-
Ranjan et al. [62]/2019	CC	-	1.2	-
Casser et al. [37]/2019	Struct2Depth	-	1.1	-
Godard et al. [63]/2019	Monodepth2	-	1.6	-
Luo et al. [65]/2020	EPC++	-	1.2	-
Zhu et al. [130]/2021	NeuralR-Pose	-	-	21
Lee et al. [66]/2021	Insta-DM	-	1.05	-
Zou et al. [128]/2022	ObjectFusion_S3	0.79	-	-
Zou et al. [128]/2022	ObjectFusion_S1	0.964	-	-
Qiao et al. [131]/2022	ORGPoseNet	-	-	21
Zou et al. [128]/2022	ObjectFusion_S4	1.132	-	-
Qiao et al. [131]/2022	ORGMapNet	-	-	20
Mumuni et al. [48]/2022	Cowan	-	1.15	-
Mumuni et al. [48]/2022	Cowan-GGR	-	1.05	-

where, d_p^{gt} and d_p are the ground-truth depth and estimated depth of pixel p, respectively.

Displacement error (DE - e^t) is the displacement error of the object compared to the ground-truth position. Rotation error (RE - e^r) is the object's rotation angle error compared to the ground-truth data.

Results of Visual SLAM when using the DL map construction module are presented in Table VIII. Similar to the previous modules, the map construction based on the DL module is also evaluated on many datasets and with many different measures. Many cells in Table VIII are empty. The above results are that the smaller the better.

e. DL Loop Closure Detection Module

Hou et al. [150] proposed a pre-trained CNN model for creating an appropriate image representation to detect visual loop closure. The pre-trained CNN model is trained from more than 2.5 million images of 205 scene categories of the scene-centric dataset. It is easy to extract CNN whole-image descriptors, and then select the most suitable layer for detecting Visual SLAM's loop closure.

Xia et al. [151] proposed to use and compare several DL networks to detect loop closure in the Visual SLAM framework as PCANet, CaffeNet, AlexNet, and GoogLeNet.

Zhang et al. [152] proposed using a pre-trained CNN model to generate whole-image descriptors for loop closure detection. To detect loop closure, the CNN model performs similarity matrix calculation. The architecture of the CNN model includes convolution, max-pooling operation, and a fully connected layer with an input image size of 221×221, and the output is a vector with more than 1000 elements.

Merrill et al. [153] proposed an unsupervised DL model with convolutional auto-encoder network architecture. The proposed network used the HOG feature on the training data, thereby creating a compact and lightweight model for real-time loop closing.

Memon et al. [154] proposed a method using two DL networks to detect the loop closure detection more accurately. The proposed method ignores coarse features such as moving objects in the environment such as cycles, bikes, pedestrians, vehicles, and any animals. To extract deep features, the

TABLE VIII. RESULTS OF VISUAL SLAM WHEN USING THE DL MAP CONSTRUCTION MODULE

Authors/ Years	Methods/ Datasets/ Measu.	KITTI 2012 dataset [39]		NYU RGB-D V2 dataset [42]			TUM RGB-D SLAM dataset [45]			ICL-NUIM dataset [46]			Mask-RCNN MC dataset [140]	
		t_{rel} (%)	r_{rel} (°)	RMSE	ALE (log)	AbRE (abs. rel)	RMSE	ALE (log)	AbRE (abs. rel)	RMSE	ALE (log)	AbRE (abs. rel)	DE	RE
Geiger et al. [147]/ 2011	VISO-S	2.05	1.19	-	-	-	-	-	-	-	-	-	-	-
Geiger et al. [147] /2011	VISO-M	19	3.23	-	-	-	-	-	-	-	-	-	-	-
Haarnoja et al. [148] /2016	BKF	18.04	5.56	-	-	-	-	-	-	-	-	-	-	-
Liu et al. [54]/ 2016		-	-	0.73	0.33	0.33	0.86	0.29	0.25	0.81	0.41	0.45	-	-
Liu et al. [54]/ 2016	+ Fusion	-	-	0.65	0.3	0.29	0.81	0.28	0.24	0.64	0.32	0.34	-	-
Laina et al. [55]/ 2016		-	-	0.51	0.22	0.18	1.07	0.39	0.25	0.54	0.28	0.23	-	-
Laina et al. [55]/ 2016	+ Fusion	-	-	0.44	0.19	0.16	0.91	0.32	0.22	0.41	0.23	0.19	-	-
Coskun et al. [149]/ 2017	LSTM-KF	3.24	1.55	-	-	-	-	-	-	-	-	-	-	-
Coskun et al. [149]/ 2017	LSTMs	3.07	1.38	-	-	-	-	-	-	-	-	-	-	-
Zhao et al. [139]/ 2019	LKN	1.79	0.87	-	-	-	-	-	-	-	-	-	-	-
Ye et al. [47]/ 2020	DRM-SLAM_C	-	-	0.5	0.19	0.16	0.7	0.28	0.2	0.36	0.18	0.16	-	-
Ye et al. [47]/ 2020	F w/o Confidence	-	-	0.48	0.2	0.16	0.67	0.26	0.18	0.35	0.17	0.16	-	-
Ye et al. [47]/ 2020	DRM-SLAM_F	-	-	0.44	0.16	0.09	0.62	0.23	0.1	0.3	0.13	0.14	-	-
Tao et al. [140]/ 2020	Non-semantic maps without moving objects	-	-	-	-	-	-	-	-	-	-	-	0.0068 ± (0.0029)	0.0138 ± (0.0057)
Tao et al. [140]/ 2020	Semantic maps without moving objects	-	-	-	-	-	-	-	-	-	-	-	0.0045 ± (0.0029)	0.0127 ± (0.0057)
Tao et al. [140]/ 2020	Non-semantic maps with moving objects	-	-	-	-	-	-	-	-	-	-	-	0.0071 ± (0.0029)	0.0145 ± (0.0057)
Tao et al. [140]/ 2020	Semantic maps with moving objects	-	-	-	-	-	-	-	-	-	-	-	0.0057 ± (0.0029)	0.0134 ± (0.0057)

proposed method uses the VGG16 architecture and uses five convolution layers, 4 Max pooling layers, and two dense layers. As a result, the proposed method has eight times more loop closure detection accuracy than traditional feature extraction.

Chang et al. [155] proposed a triple loss-based metric learning method to embed into the Visual SLAM system to increase the accuracy of closed-loop detection. This method converted the keyframes into feature vectors, evaluating the similarity of the keyframes by calculating the Euclidean distance of feature vectors. Features on keyframes are extracted using ResNet_V1_50 with the average-pooling output size of $2048 \times 1 \times 1$. The fully connected layer is $2048 \times 1024 \times 128$.

Duan et al. [156] proposed a deep feature matching-based keyframe retrieval method to perform loop closure detection in the semantic Visual SLAM system, called DFM (deep feature matching), this method is based on the CNN method. The matching of the implementation method's current scenes with the recorded keyframes and finding the transformation between the matched keyframes for trajectory correction by matching the local pose graphs. This method converted the keyframe descriptors and pose graphs into a sparse image, with each keyframe into a feature point.

City Centre [157] was collected on a road near the city center with many moving objects such as people and vehicles, in

environmental conditions with a lot of sun and wind causing the tree shadows to change a lot. The data was collected on a road with a total length of 2km and 2,474 images were collected, with each data collection point marked in yellow, two images collected at the same location marked in red and connected by one line.

Gardens Point Walking (GPW) [153] was collected while traveling three times on a road at the QUT campus in Brisbane, Australia. This dataset shows large differences in view direction, dynamic objects, occlusions, and illumination of each pass through this path. Of the three walks on this road, two are done in one day with one walking on the left-hand side and one time on the right-hand side of pedestrians. In which, the i^{th} image in this sequence is matched with any i^{th} image in the other two sequences.

To evaluate the performance of the loop closure detection using the DL module in the Visual SLAM system, the researchers used several RGB-D datasets as shown below. AUC (Area Under the Curve) is an aggregate measure of the performance of a binary classifier across all possible threshold values. ROC curve is a curve that represents the classification performance of a classification model at thresholds. Essentially, True Positive Rate (TPR) vs False Positive Rate (FPR) for different threshold values. TPR and FPR values are calculated as formula (11).

$$TPR = \frac{TP}{TP + FP}; FPR = \frac{FP}{TN + FN} \tag{11}$$

AUC is an index calculated based on the ROC curve to evaluate how well the model can classify. The area under the ROC curve and on the horizontal axis is the AUC, with a value in the range $[0, 1]$. RMSE has been presented above. The mean of the trajectory error (MTE) is defined in formula (5). Average (Avg.) Good Match (%) is the good matches (inliers) rate of the pose graphs. Results of Visual SLAM methods when using the DL loop closure detection module are presented in Table IX. The studies were evaluated on the KITTI 2012 [39], City Centre [157], GPW [153], TUM RGB-D SLAM [45] datasets with the AUC, MTE, RMSE, and Avg. Good Match measures. With measures AUC and Avg. Good Match, the better the results are, with measures MTE and RMSE, the smaller the results are the better.

f. DL Others Module

Camera re-localization is the process of estimating the camera's location and orientation in the data collection environment using the images of the captured environment as the input. To evaluate the performance of this module, studies often evaluate based on two metrics: Angular (Ang.) error (degree) and Translation (Trans.) error (m). Some research results of the camera re-localization module based on DL on the 7-Scenes [133] dataset are shown in Table X.

Another DL-based module is distance estimation. The results of studies performing distance estimation based on image data obtained from the environment are shown in Table XI. The results are evaluated on the KITTI 2012 dataset [39] with the following measurements: $RMSE$ has been presented above; $AccDev$ is the accuracy with one-meter deviation; Acc is more accurate than the accurate one. In which the $RMSE$ result is as small as possible, the larger Acc and $DevAcc$ are the better.

Another DL-based module is scene reconstruction. The results of studies performing 3D reconstruction scenes based on image data (RGB-D) obtained from the environment are shown in Table XI. In Table XI, 3D reconstruction scene methods are based on RGB or depth images of NYU RGB-D V2 [42], KITTI 2012 [39] in Table XII, and Make3D [43] datasets. When using RGB images as input, methods often use the method of estimating the depth of the image, and then combining it with color images to build a 3D scene with point cloud data. These studies often perform and improve image depth estimation models.

5) End-to-end for the Visual SLAM Algorithm

As presented in Fig. 1 and Fig. 2, the research based on the DL for Visual SLAM and VO systems is very diverse. The research can only be applied to one module of the Visual SLAM and VO framework. Currently, most research using the end-to-end DL methods is mainly used for the VO construction process with the basic output being the camera's moving trajectory/camera pose (6 DOF) in the environment. Weber et al. [194] proposed a CNN for extracting and training temporal features on videos using a Slow Fusion Network and Early Fusion Network with dimensions of ($390 \times 130 \times 10 \times 3$), ($390 \times 130 \times 2 \times 3$) to estimate ego-motion. Slow Fusion Network has input data of 10 consecutive frames of a video and uses up to 5 conv. layers, Early Fusion Network uses the input of 2 consecutive frames of a video and all convolution layers are 2D convolution. Wang et al. [195] proposed an end-to-end framework, called DeepVO for VO estimation, the process of extracting the conventional feature-based monocular VO is based on CNN, the process of learning the CNN features extracted from motion information and estimating poses of two consecutive monocular RGB images is by using RCNN. Peretroukhin et al. [196] proposed a model called Sun-BCNN (Sun Bayesian Convolutional Neural Network) for estimating VO, in which Bayesian CNN is used to detect the direction of the sun from an RGB image using global orientation information as a mean and covariance. The final VO is built upon a sliding window bundle adjuster. Li et al. [197] proposed an unsupervised DL, called UnDeepVO for VO estimation. The input data to train the model is stereo image pairs, the features are extracted from both spatial and temporal geometric constraints, but the model can perform estimating VO, 6-DOF poses, and depth estimation with monocular images. To estimate pose, UnDeepVO uses features extracted from VGG. UnDeepVO

TABLE IX. RESULTS OF VISUAL SLAM METHODS WHEN USING THE DL LOOP CLOSURE DETECTION MODULE

Authors/ Years	Datasets/ measurement\ Methods	KITTI 2012 dataset [39] (00,02 ,05)	City Centre [157]	GPW [153]	TUM RGB-D SLAM dataset [45] (fr1_desk, fr2_desk, fr3_ long _office)		KITTI 2012 dataset [39] (00,02,08)		KITTI 2012 dataset [39] (00)
		AUC	AUC	AUC	MTE	RMSE	MTE	RMSE	Avg. Good Match (%)
Galvez et al. [158]/2012	DBoW2_ORB	0.067	0.22	0.092	-	-	-	-	-
Galvez et al. [158]/2012	DBoW2_BRISK	0.318	0.186	0.088	-	-	-	-	-
Galvez et al. [158]/2012	DBoW2_SURF	0.175	0.177	0.086	-	-	-	-	-
Galvez et al. [158]/2012	DBoW2_AKAZE	0.413	0.444	0.199	-	-	-	-	-
Rmsa et al. [159]/2017	DBoW3_ORB	0.274	0.217	0.182	-	-	-	-	-
Rmsa et al. [159]/2017	DBoW3_BRISK	0.169	0.187	0.098	-	-	-	-	-
Rmsa et al. [159]/2017	DBoW3_SURF	0.12	0.019	0.0197	-	-	-	-	-
Rmsa et al. [159]/2017	DBoW3_AKAZE	0.46	0.174	0.147	-	-	-	-	-
Garcia et al. [160]/2018	iBoW	0.88	0.94	0.95	-	-	-	-	-
Sarlin et al. [161]/2019	HF-Net	-	-	-	-	-	-	-	-
Memon et al. [154]/2020	Impro_BoW _Without AE	0.912	0.96	0.94	-	-	-	-	-
Memon et al. [154]/2020	Impro_BoW _With AE	0.96	0.97	0.97	-	-	-	-	-
Chang et al. [155]/2021	Triplet Loss _BoW	-	-	-	0.014	0.016	5.416705	6.74	
Chang et al. [155]/2021	Triplet Loss _Metric _Learning	-	-	-	0.012	0.0135	2.92	3.46	
Duan et al. [156]/2022	CNN_DFM	-	-	-	-	-	-	-	63

TABLE X. RESULTS OF VISUAL SLAM METHODS WHEN USING THE DL CAMERA RE-LOCALIZATION MODULE

Authors/Years	Dataset/ Measu. Methods	7-Scenes dataset	
		Ang. error (degree)	Trans. error (m)
Kendall et al. [162], [163]/2016	PoseNet	10.4	0.44
Kendall et al. [162], [163]/2016	Bayesian PoseNet	9.81	0.47
Kendall et al. [162], [163]/2016	PoseNet-Euler6	9.83	0.38
Kendall et al. [162], [163]/2016	PoseNet-Euler6-Aug	8.58	0.34
Kendall et al. [162], [163]/2016	BranchNet-Euler6	9.82	0.3
Kendall et al. [162], [163]/2016	BranchNet-Euler6-Aug	8.3	0.29
Kendall et al. [137]/2017	Geometric PoseNet	8.1	0.23
Melekhov et al. [164]/2017	Hourglass	9.5	0.23
Walch et al. [165]/2017	LSTM-Pose	9.9	0.31
Wu et al. [166]/2017	BranchNet	8.3	0.29
Wang et al. [167]/2019	MLFBPPose	9.8	0.2
Bui et al. [168]/2019	ANNet	7.9	0.21
Cai et al. [169]/2019	GPoseNet	10.0	0.31
Saha et al. [170]/2019	AnchorPoint	7.5	0.13
Wang et al. [171]/2020	AttLoc	7.6	0.2
Turkoglu et al. [172]/2021	GNN-RPS	5.2	0.16

uses an encoder-decoder architecture to generate dense depth maps. Zhan et al. [198] proposed a Depth-VO-Feat framework to estimate depth images using CNN and other CNN-based VO from stereo sequences. Depth-VO-Feat framework can estimate single-view depths and two-view odometry which can reduce scaling ambiguity issues. Shamwell et al. [199] propose a Visual-Inertial-Odometry Learner (VIOLearner) method based on an unsupervised deep neural network to combine RGB-D images and inertial measurement unit (IMU) intrinsic parameters of the camera to estimate the camera's moving trajectory/camera pose in the environment. IMU data is fed through CNN layers and the output is a 3D Affine matrix that estimates the change in camera pose between a source image and a target image. VIOLearner uses the input data including an RGB-D source image, a target RGB image, IMU data, and a camera calibration matrix K with the camera's intrinsic parameters. VIOLearner generates hypothesis trajectories and then corrects them online according to the Jacobians of the

TABLE XI. RESULTS OF VISUAL SLAM METHODS WHEN USING THE DL DISTANCE ESTIMATION MODULE

Authors/Years	Dataset/ Measu. Methods	KITTI 2012 dataset [39] (03, 04, 05, 06, 07,10)			KITTI 2012 dataset [39] (09,10)		
		RMSE	Acc	AccDev	RMSE	Acc	AccDev
Mur et al. [104]/2015	ORB-SLAM-mono	7.4623	0.0221	0.0368			
Engel et al. [173] /2016	DSO-mono	7.3854	0.0241	0.0452			
Fanani et al. [174]/2017	PMO	0.7463	0.7183	0.9633			
Wang et al. [175]/2017	DSO-stereo	0.0756	0.9387	1			
Yin et al. [60]/2018	GeoNet				6.2302	0.0306	0.0544
Xue et al. [176]/2019	SRNN	0.6754	0.6121	0.9667			
Xue et al. [176]/2019	SRNN-se	0.6526	0.5801	0.9727			
Xue et al. [176]/2019	SRNN-point	0.5234	0.6267	0.9822			
Xue et al. [176]/2019	SRNN-channel	0.5033	0.6487	0.9873			
Kreuzig et al. [177]/2019	DistanceNet-FlowNetS	0.5544	0.6292	0.9752			
Kreuzig et al. [177]/2019	DistanceNet-Reg	0.5315	0.6848	0.9855			
Kreuzig et al. [177]/2019	DistanceNet-LSTM	0.4167	0.6871	0.9896			
Kreuzig et al. [177]/2019	DistanceNet-BCE	0.3925	0.7158	0.993			
Kreuzig et al. [177]/2019	DistanceNet	0.3901	0.6984	0.9916	0.4624	0.6669	0.9841
Bian et al. [38]/2019	SfMLearner				7.5671	0.0216	0.0505

TABLE XII. THE RESULTS OF ESTIMATING THE CAMERA'S MOVING TRAJECTORY ON THE IMAGE SEQUENCES OF KITTI 2012 DATASET [39]

Authors/Years	Methods	Dataset/ Measu./ Output	KITTI 2012 dataset [39] (00, 02, 05, 07, 08)		KITTI 2012 dataset [39] (09, 10)		KITTI 2012 dataset [39] (03, 04, 05, 06, 07,10)	
			$t_{rel}(\%)$	$r_{rel}(°)$	$t_{rel}(\%)$	$r_{rel}(°)$	$t_{rel}(\%)$	$r_{rel}(°)$
Leutenegger et al. [202]/2015	OKVIS	Trajectory estimation	-	-	13.535	2.895	-	-
Zhou et al. [32]/2017	SFMLearner	Trajectory estimation	36.232	4.562	21.085	7.25	-	-
Bloesch et al. [203]/2017	ROVIO	Trajectory estimation	-	-	20.11	2.165	-	-
Wang et al. [195] /2017	DeepVO	Trajectory estimation	-	-	-	-	5.96	6.12
Shamwell et al. [199]/2018	VIOLearner	Trajectory estimation	5.574	2.31	1.775	1.135	-	-
Li et al. [197]/2018	UnDeepVO	Trajectory estimation	4.07	2.026	-	-	-	-
Li et al. [197]/2018	VISO2-M	Trajectory estimation	17.924	2.798	-	-	17.48	16.52
Li et al. [197]/2018	ORB-SLAM-M	Trajectory estimation	27.0575	10.2375	-	-	-	-
Li et al. [197]/2018	VISO2-M	Trajectory estimation	-	-	-	-	1.89	1.96
Zhan et al. [198]/2018	Depth-VO-Feat	Trajectory estimation	-	-	12.27	3.52	-	-
Almalioglu et al. [201]/2022	SelfVIO	Trajectory estimation	0.9	0.44	1.88	1.23	-	-
Almalioglu et al. [201]/2022	SelfVIO (no IMU)	Trajectory estimation	-	-	2.41	1.62	-	-
Almalioglu et al. [201]/2022	SelfVIO (LSTM)	Trajectory estimation	-	-	2.07	1.32	-	-

error image obtained with the original coordinates. Yang et al. [200] proposed a DL framework, called D3VO for building VO with three levels: deep depth, pose, and uncertainty estimation. The first level is to use a self-supervised network to estimate depth from stereo videos using DepthNet from a single image, the second level is to estimate the pose between adjacent images using PoseNet, and the third level is to estimate the associated uncertainty. Incorporating temporal information into the depth estimation learning process. Almalioglu et al. [201] proposed SelfVIO (self-supervised DL-based VIO) to estimate the camera's moving trajectory and depth from input data of monocular RGB image sequences and IMU. SelfVIO can perform estimating the relative translation and rotation between consecutive frames parametrized as 6 DOF motion and a depth image. To recover the camera's movement trajectory in the environment, SelfVIO used the convolutional layers. Studies based on end-to-end DL for building VO systems/estimating trajectory motion/camera pose in the environment often use metrics t_{rel} and r_{rel} to evaluate the results, which have been presented above. The results based on these two measures are that lower is better. The results of estimating the camera's moving trajectory in the environment based on end-to-end DL on the image sequences of the KITTI 2012 dataset [39] are shown in Table XII. In Table XII, the research results of end-to-end DL have also been compared with traditional ML methods such as ORB-SLAM-M, and the results show that the methods using end-to-end DL are better than traditional ML.

IV. CHALLENGES AND DISCUSSIONS

Visual SLAM and VO systems are applied and are very important components in building robot systems, autonomous mobile robots, and assistance systems for the blind, human-machine interaction, industry, etc. Based on the above surveys, it can be seen that the results of Visual SLAM and VO systems have been significantly improved when using DL in the system modules or the end-to-end DL systems. As presented in Fig. 1 and Fig. 2, the Visual SLAM and VO systems must perform through many steps, there may be many intermediate results

in each step, so there are many challenges that need to be resolved to have a good Visual SLAM and VO systems. During the survey of research on Visual SLAM and VO systems, we realized that there are some challenges and discussions in this problem on the RGB-D images which are specifically presented as follows.

A. *Performances of Visual SLAM and VO systems*

DL has delivered convincing results in the Visual SLAM and VO systems. However, DL is a method based on statistical ML, therefore, the results in the steps of the Visual SLAM and VO system-building process all have certain errors. Based on the model illustrated in Fig. 1, the errors can accumulate and the output has very large errors concerning the original data. To minimize these errors, end-to-end models were built based on DL. However, the accuracy was only partially improved, the results are presented in Table XIII. Most of the studies using DL to build the Visual SLAM and VO systems often exploit environment features based on the RGB-D images obtained from the environment. The features extracted using DL are mainly space. When moving in the environment, the data obtained from the environment is usually a sequence of frames. Therefore, temporal features need to be researched and extracted to improve the performance of environmental map construction, shown in Fig. 3.

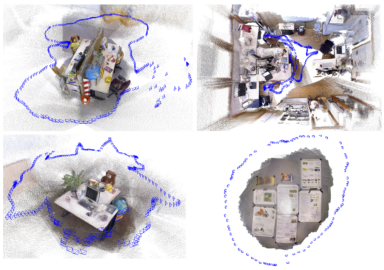

Fig. 3. Environment map built using ORB-SLAM 2.0 [102] on TUM RGB-D dataset

Another issue concerns the performance of DL methods, with 3D real-world spaces containing many environmental challenges such as environmental complexity, moving objects, lighting, etc. These factors all affect the performance of the learning model. Therefore, with the DL method of learning the environment, the methods often have to use supervised learning methods to learn features extracted from the environment such as studies [156], [204], [205], [206], [195], [207]. With the methods of using unsupervised data, they are often performed in very specific and uniform environmental conditions such as studies [208], [199], [197], [201], [209].

B. *Energy Consumption and Computing Space*

It can be seen that DL networks have brought impressive results for building Visual SLAM and VO systems. However, whether DL methods are used at one step in the Visual SLAM and VO systems building model or end-to-end, DL is usually computed on the GPU. To equip GPUs, the cost is more expensive than CPUs, and other devices, especially GPUs consume a much larger amount of power than CPUs. Visual SLAM and VO systems are typically installed on CPU-only computers or edge devices. These computers can be mounted on moving robots, industrial autonomous vehicles, self-driving cars, etc. Therefore, the power supply for these computer devices is relatively limited. Although, recently there have been some studies on building Visual SLAM and VO systems by computing on edge devices/split computing on several devices such as [210], [211], [212], [213]. However, these studies are still only tested in the laboratory. Another issue of computational space is that when constructing the Visual SLAM and VO systems in a large space shown in Fig. 4, the computational space will increase according to the number of frames obtained from the environment if the amount of data obtained gradually reduces the calculation speed and reaches a threshold that will overflow the computer's memory. Especially in the case of building environment maps and 3D scene reconstruction.

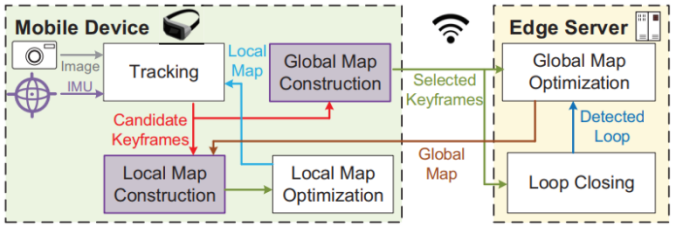

Fig. 4. AdaptSLAM's design model performs distributed computing on edge devices [214].

C. *Generalize and Adaptive*

Although, currently research using DL methods to build Visual SLAM and VO systems has quite convincing results. However, the learning methods using DL networks are mainly learned from fixed scenes and environments, with little mixing and few moving objects. Learn-based methods often exploit well the features extracted from the environment, subjects should have good results in the learning environment. When moving to a highly moving environment with a few more objects, the effectiveness of the learned model is no longer maintained. For example, building an environment map for autonomous vehicles moving in a factory. During the process of moving in the environment, suddenly another object moves onto the path, and the environment of the autonomous vehicle has been learned, which causes the environment to change and the autonomous vehicle can not complete the job due to the wrong path estimation. Therefore, the issue of environmental generalization and adaptation in environmental conditions with mixed flutes needs to be studied further. From there, it is possible to build an environment learner that meets many situations when

TABLE XIII. RESULTS OF VISUAL SLAM METHODS WHEN USING THE DL SCENE RECONSTRUCTION MODULE

Authors/Years	Dataset/ Measu./ Methods	NYU RGB-D V2 dataset [42]		NYU RGB-D V2 dataset [42]		KITTI 2012 dataset [39]		Make3D [43] dataset	
		RGB		Depth		RGB		RGB	
		RMSE	REL	RMSE	REL	RMSE	REL	RMSE	REL
Saxena et al. [178]/2008	Samples_0	-	-	-	-	-	-	16.7	0.53
Saxena et al. [43]/2009	Samples_0	-	-	-	-	8.374	0.28	-	0.698
Eigen et al. [30]/2014	Samples_0	-	-	-	-	7.156	0.19	-	-
Eigen et al. [53]/2015	Samples_0	0.641	0.158	-	-	-	-	-	-
Laina et al. [55]/2016	Samples_0	0.573	0.127	-	-	-	-	-	-
Roy et al. [179]/2016	Samples_0	0.744	0.187	-	-	-	-	-	-
Mancini et al. [180]/2016	Samples_0	-	-	-	-	7.508	-	-	-
Cadena et al. [181]/2016	Samples_650	-	-	-	-	7.14	0.179	-	-
Xu et al. [182]/2017	Samples_0	0.586	0.121	-	-	-	-	-	-
Liao et al. [183]/2017	Samples_225	0.442	0.104	-	-	-	-	-	-
Liao et al. [183]/2017	Samples_225	-	-	-	-	4.5	0.113	-	-
Xu et al. [184]/2018	Samples_0	0.593	0.125	-	-	-	-	-	-
Xu et al. [184]/2018	Samples_0	0.582	0.12	-	-	-	-	-	-
Mal et al. [56]/2018	Samples_0	0.514	0.143	-	-	6.266	0.208	-	-
Mal et al. [56]/2018	Samples_20	0.351	0.078	0.461	0.11	-	-	-	-
Li et al. [185]/2018	(L2 loss)	0.943	0.572			-	-	-	-
Li et al. [185] /2018	L1 loss	0.256	0.046	0.68	0.24	-	-	-	-
Mal et al. [56]/2018	Samples_200	0.23	0.044	0.259	0.054	-	-	-	-
Li et al. [185]/2018	L1 loss Samples_50	-	-	0.44	0.13	-	-	-	-
Mal et al. [56]/2018	Samples_50	-	-	0.347	0.076	-	-	-	-
Mal et al. [56]/2018	Samples_500	-	-	-	-	3.378	0.073	5.525	0.14
Li et al. [185]/2018	L1 loss samples_200	-	-	0.39	0.1	-	-	-	-
Wang et al. [186]/2018	Samples_0	-	-	-	-	6.298	0.18	-	-
Wofk et al. [187]/2019	Samples_0	0.583	0.164	-	-	5.191	0.145	10.281	0.594
Gur et al. [188]/2019	Samples_0	0.766	0.254	-	-	5.187	0.141		
Xu et al. [189]/2019	Samples_0	0.579	0.108	-	-	-	-	-	-
Tu et al. [190]/2019	Samples_0	0.547	0.152	-	-	-	-	-	-
Wang et al. [191]/2019	Samples_100	0.502	-	-	-	-	-	-	-
Hu et al. [192]/2019	Samples_20	0.526	-	1.369	-	-	-	-	-
Wofk et al. [187]/2019	Samples_20	0.385	0.086	0.462	0.106	-	-	-	-
Hu et al. [192]/2019	Samples_200	0.495	-	1.265	-	-	-	-	-
Wofk et al. [187]/2019	Samples_200	0.292	0.068	0.289	0.062	-	-	-	-
Tu et al. [190]/2019	Samples_20	-	-	0.457	0.107	-	-	-	-
Hu et al. [192] /2019	Samples_50	-	-	1.31	-	-	-	-	-
Wofk et al. [187]/2019	Samples_50	-	-	0.35	0.075	-	-	-	-
Hu et al. [192]/2019	Samples_0	-	-	-	-	5.437	-	-	-
Hu et al. [192]/2019	Samples_500	-	-	-	-	5.389	-	-	-
Wang et al. [191]/2019	Samples_500	-	-	-	-	5.14	-	-	-
Wofk et al. [187]/2019	Samples_500	-	-	-	-	3.033	0.051	5.658	0.135
Tu et al. [193]/2020	DEM_ samples_0	0.49	0.135	-	-	4.433	0.101	10.003	0.529
Tu et al. [193] /2020	w/o pre-trained weights samples_0	0.637	0.187	-	-	-	-	-	-
Tu et al. [193]/2020	DEM_samples_20	0.314	0.069	0.443	0.1	-	-	-	-
Tu et al. [193]/2020	DEM_ samples_200	0.194	0.036	0.223	0.041	-	-	-	-
Tu et al. [193]/2020	w/o pre-trained weights	0.226	0.042	0.23	0.043	-	-	-	-
Tu et al. [193]/2020	DEM_ samples_50	-	-	0.342	0.07	-	-	-	-
Tu et al. [193]/2020	DEM_ samples_500	-	-	-	-	2.485	0.04	5.455	0.104

moving, lighting changes, objects in the environment change, etc. The issue of evaluating the results of Visual SLAM and VO systems is only relative, as shown in Table II, Table III, Table IV, Table V, Table VI, Table VII, Table VIII, Table IX, Table X, Table XI, Table XII, Table XIII. Although, using the DL method in the same step or from beginning to end of the system, the methods are different. Different evaluations are performed on different measures. Therefore, this issue needs to be unified so that peer comparison between methods ensures effectiveness.

D. Actual Implementation

It can be seen that the biggest drawback of ML models as well as the DL method is the problem of changing the environment between training and experimentation. However, we have tried to learn all the situations that can occur in reality and tried to build an experimental environment close to the real environment. However, this is never enough, especially since many unusual situations arise in reality, changing lighting conditions, and timing causes the environment to change. These changes may not be learned by the DL models or not learned much. The results of building the environmental map of the Visual SLAM and VO systems are not good, making the actual implementation difficult and yielding poor results. Although, the studies of Visual SLAM and VO systems have been evaluated on multiple datasets such as KITTI 2012 [39], KITTI 2015 [41], NYU RGB-D V2 [42], 7-Scenes [133], TUM RGB-D SLAM [45], GPW [153], ICL-NUIM [46], Mask-RCNN MC [140], SceneNet RGB-D [132], CARLA [125], Object Fusion [113], ADVIO [28], Euroc [105], Sintel Final [80], Middlebury [81], Flying Chairs [67], Foggy [78], etc. Many different lighting conditions, indoors or outdoors, performed at many different times, etc. The results have been shown and

compared in Table II, Table III, Table IV, Table V, Table VI, Table VII, Tab. VIII, Table IX, Tab. X, Table XI, Table XIII, and Table XII. However, the issue of practical implementation of Visual SLAM and VO systems still faces many challenges and needs further research.

Currently, to deploy the Visual SLAM and VO applications into practice, we use RGB-D image information obtained from the environment. The system needs to have a calculation time equal to or close to reality (about 24fps). To build a system with such computing time, the system needs to perform calculations on a relatively large GPU. Therefore, integrating GPUs onto embedded computers and mobile devices is also a challenge that needs further research.

V. Conclusions and Future works

Visual SLAM and VO systems are often the core of control systems on autonomous vehicle systems, industrial robots, guidance systems, etc. The advent and strong development of DL methods have brought very impressive results in solving ML and computer vision problems with RGB-D image as the input data. In this paper, we proposed the taxonomy for investigating DL-based methods to perform Visual SLAM and VO methods from RGB-D images sensors. We have also conducted a complete survey of more than 200 studies on building Visual SLAM, VO, and related systems. This survey is based on three main directions of the Visual SLAM and VO systems: applying DL modules to the steps of Visual SLAM and VO systems, replacing the DL modules to the Visual SLAM and VO systems, applying end-to-end DL to the Visual SLAM and VO systems. In which the studies are presented in order of methods, evaluation datasets, evaluation measures, and experimental results. The results show that, despite receiving a lot of research attention in the past 10 years, the results on the Visual SLAM and VO systems are scattered according to many different criteria, because each study may only focus on one step of the Visual SLAM and VO system. At the same time, we also present discussions and challenges to build Visual SLAM and VO systems.

Shortly, we will build a standard dataset to evaluate Visual SLAM and VO with full contexts and situations that can occur in reality when blind people move and find their way into the kitchen room. At the same time, try to improve the process of building moving maps and finding directions for blind people in the kitchen, especially in new environments.Conducting a comparative study on using DL to estimate VO/camera pose/camera trajectory on the self-constructed and collected data.

Acknowledgment

This research is funded by Tan Trao University in Tuyen Quang province, Vietnam.

References

[1] J. Zhao and S. Liu, "Research and Implementation of Autonomous Navigation for Mobile Robots Based on SLAM Algorithm under ROS," *Sensors*, vol. 22, 2022, doi: 10.3390/s22114172.

[2] P. V. Dhayanithi and T. Mohanraj, "Ros-based evaluation of slam algorithms and autonomous navigation for a mecanum wheeled robot," in *Intelligent Control, Robotics, and Industrial Automation*, pp. 13–24, 2023, doi: 10.1007/978-981-99-4634-1_2.

[3] P. T. Karfakis and M. S. Couceiro, "NR5G-SAM : A SLAM Framework for Field Robot Applications Based on 5G New Radio," *Sensors*, vol. 23, pp. 1–33, 2023, doi: 10.3390/s23115354.

[4] S. Zheng, J. Wang, C. Rizos, W. Ding, and A. El-mowafy, "Simultaneous Localization and Mapping (SLAM) for Autonomous Driving : Concept and Analysis," *Remote Sensing*, vol. 15 no. 4, pp. 1–41, 2023, doi: 10.3390/rs15041156.

[5] M. Bamdad, D. Scaramuzza, and A. Darvishy, "SLAM for Visually Impaired People : a Survey," *IEEE Access*, vol. 11, pp. 1–45, 2023, doi: 10.48550/arXiv.2212.04745.

[6] W. Kontar, S. Ahn, Y. Liu, M. Tight, and Y. Gong, "Research on Comparison of LiDAR and Camera in Autonomous Driving Research on Comparison of LiDAR and Camera in Autonomous Driving," *Journal of Physics: Conference Series*, vol. 2093, 2021, doi: 10.1088/1742-6596/2093/1/012032.

[7] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 6, 2017, doi: 10.1186/s41074-017-0027-2.

[8] L. Jinyu, Y. Bangbang, C. Danpeng, W. Nan, Z. Guofeng, and B. Hujun, "Survey and evaluation of monocular visual-inertial SLAM algorithms for augmented reality," *Virtual Reality and Intelligent Hardware*, vol. 1, no. 4, pp. 386–410, 2019, doi: 10.1016/j.vrih.2019.07.002.

[9] D. Lai, Y. Zhang and C. Li, "A Survey of Deep Learning Application in Dynamic Visual SLAM," *2020 International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE)*, pp. 279-283, 2020, doi: 10.1109/ICBASE51474.2020.00065.

[10] R. Azzam, T. Taha, S. Huang, and Y. Zweiri, "Feature-based visual simultaneous localization and mapping: a survey," *SN Applied Sciences*, vol. 2, no. 2, pp. 1–24, 2020, doi: 10.1007/s42452-020-2001-3.

[11] L. Xia, J. Cui, R. Shen, X. Xu, Y. Gao, and X. Li, "A survey of image semantics-based visual simultaneous localization and mapping: Application-oriented solutions to autonomous navigation of mobile robots," *International Journal of Advanced Robotic Systems*, vol. 17, no. 3, pp. 1–17, 2020, doi: 10.1177/1729881420919185.

[12] B. Fang, G. Mei, X. Yuan, L. Wang, Z. Wang, and J. Wang, "Visual SLAM for robot navigation in healthcare facility," *Pattern Recognition*, vol. 113, 2021, doi: 10.1016/j.patcog.2021.107822.

[13] A. M. Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A Comprehensive Survey of Visual SLAM Algorithms," *Robotics*, vol. 11, no. 1, 2022, doi: 10.3390/robotics11010024.

[14] I. Abaspur Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual SLAM," *Expert Systems with Applications*, vol. 205, 2022, doi: 10.1016/j.eswa.2022.117734.

[15] J. Qin, M. Li, D. Li, J. Zhong, and K. Yang, "A Survey on Visual Navigation and Positioning for Autonomous UUVs" *Remote Sensing*, vol. 14, no. 15, 2022, doi: 10.3390/rs14153794.

[16] Z. Zhang and J. Zeng, "A Survey on Visual Simultaneously Localization and Mapping," *Frontiers in Computing and Intelligent Systems*, vol. 1, no. 1, pp. 18–21, 2022, doi: 10.54097/fcis.v1i1.1089.

[17] K. A. Tsintotas, L. Bampis and A. Gasteratos, "The Revisiting Problem in Simultaneous Localization and Mapping: A Survey on Visual Loop Closure Detection," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 19929-19953, 2022, doi: 10.1109/TITS.2022.3175656.

[18] K. Chen, J. Zhang, J. Liu, Q. Tong, R. Liu, and S. Chen, "Semantic Visual Simultaneous Localization and Mapping: A Survey," *arXiv*, vol. 14, no. 8, pp. 1–14, 2022, doi: 10.48550/arXiv.2209.06428.

[19] Y. Tian, H. Yue, B. Yang, and J. Ren, "Unmanned Aerial Vehicle Visual Simultaneous Localization and Mapping: A Survey," *Journal of Physics: Conference Series*, vol. 2278, no. 1, 2022, doi: 10.1088/1742-6596/2278/1/012006.

[20] A. Tourani, H. Bavley, Jose-Luis, Sanchez-Lopezz, and H. Voos, "Visual SLAM: What are the Current Trends and What to Expect?," *sensors*, vol. 22, no. 23, 2022, doi: 10.3390/s22239297.

[21] Y. Dai, J. Wu, and D. Wang, "A Review of Common Techniques for Visual Simultaneous Localization and Mapping," *Journal of Robotics*, vol. 2023, 2023, doi: 10.1155/2023/8872822.

[22] S. Mokssit, D. B. Licea, B. Guermah and M. Ghogho, "Deep Learning Techniques for Visual SLAM: A Survey," in *IEEE Access*, vol. 11, pp. 20026-20050, 2023, doi: 10.1109/ACCESS.2023.3249661.

[23] M. N. Favorskaya, "Deep Learning for Visual SLAM: The State-of-the-Art and Future Trends," *Electronics*, vol. 12, no. 9, 2023, doi: 10.3390/electronics12092006.

[24] L. R. Agostinho, N. M. Ricardo, M. I. Pereira, A. Hiolle and A. M. Pinto, "A Practical Survey on Visual Odometry for Autonomous Driving in Challenging Scenarios and Conditions," in *IEEE Access*, vol. 10, pp. 72182-72205, 2022, doi: 10.1109/ACCESS.2022.3188990.

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset. The International Journal of Robotics Research," *The International Journal of Robotics Research*, pp. 1–6, 2013.

[26] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016, doi: 10.1177/0278364915620033.

[27] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler and D. Cremers, "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1680-1687, 2018, doi: 10.1109/IROS.2018.8593419.

[28] S. Cortes, A. Solin, E. Rahtu, and J. Kannala, "ADVIO: An authentic dataset for visual-inertial odometry," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 419–434, 2018, doi: 10.1007/978-3-030-01249-6_26.

[29] C. Theodorou, V. Velisavljevic, V. Dyo, and F. Nonyelu, "Visual SLAM algorithms and their application for AR, mapping, localization and wayfinding," *Array*, vol. 15, 2022, doi: 10.1016/j.array.2022.100222.

[30] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in Neural Information Processing Systems*, vol. 3, pp. 2366–2374, 2014.

[31] W. Chen, Z. Fu, D. Yang, and J. Deng, "Single-image depth perception in the wild," *Advances in Neural Information Processing Systems*, pp. 730–738, 2016.

[32] T. Zhou, M. Brown, N. Snavely and D. G. Lowe, "Unsupervised Learning of Depth and Ego-Motion from Video," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6612-6619, doi: 10.1109/CVPR.2017.700.

[33] C. Wang, J. M. Buenaposada, R. Zhu and S. Lucey, "Learning Depth from Monocular Videos Using Direct Methods," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2022-2030, 2018, doi: 10.1109/CVPR.2018.00216.

[34] F. Steinbrücker, J. Sturm, and D. Cremers, "Real-time visual odometry from dense RGB-D images," in *Proceedings of the IEEE International Conference on Computer Vision*, no. 3, pp. 719–722, 2011, doi: 10.1109/ICCVW.2011.6130321.

[35] R. Garg, B. G. Vijay Kumar, G. Carneiro, and I. Reid, "Unsupervised CNN for single view depth estimation: Geometry to the rescue," *Computer Vision – ECCV 2016*, vol. 9912, pp. 740–756, 2016, doi: 10.1007/978-3-319-46484-8_45.

[36] C. Godard, O. M. Aodha and G. J. Brostow, "Unsupervised Monocular Depth Estimation with Left-Right Consistency," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6602-6611, 2017, doi: 10.1109/CVPR.2017.699.

[37] V. Casser, S. Pirk, R. Mahjourian and A. Angelova, "Unsupervised Monocular Depth and Ego-Motion Learning With Structure and Semantics," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 381-388, 2019, doi: 10.1109/CVPRW.2019.00051.

[38] J.-W. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-m. Cheng, and I. Reid, "Unsupervised Scale-consistent Depth and Ego-motion Learning from Monocular Video," *arXiv*, pp. 1–12, 2019.

[39] A. Geiger, P. Lenz and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence*, pp. 3354-3361, 2012, doi: 10.1109/CVPR.2012.6248074.

[40] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 23, no. 1, pp. 1–6, 2013, doi: 10.1177/0278364913491297.

[41] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3061-3070, 2015, doi: 10.1109/CVPR.2015.7298925.

[42] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor Segmentation and Support Inference from RGBD Images," *Computer Vision – ECCV 2012*, pp. 746–760, 2012, doi: 10.1007/978-3-642-33715-4_54.

[43] A. Saxena, M. Sun and A. Y. Ng, "Make3D: Learning 3D Scene Structure from a Single Still Image," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824-840, 2009, doi: 10.1109/TPAMI.2008.132.

[44] M. Cordts *et al.,* "The Cityscapes Dataset for Semantic Urban Scene Understanding," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3213-3223, 2016, doi: 10.1109/CVPR.2016.350.

[45] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573-580, 2012, doi: 10.1109/IROS.2012.6385773.

[46] A. Handa, T. Whelan, J. McDonald and A. J. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1524-1531, 2014, doi: 10.1109/ICRA.2014.6907054.

[47] X. Ye, X. Ji, B. Sun, S. Chen, Z. Wang, and H. Li, "DRM-SLAM: Towards dense reconstruction of monocular SLAM with scene depth fusion," *Neurocomputing*, vol. 396, pp. 76–91, 2020, doi: 10.1016/j.neucom.2020.02.044.

[48] F. Mumuni, A. Mumuni, and C. K. Amuzuvi, "Deep learning of monocular depth, optical flow and ego-motion with geometric guidance for UAV navigation in dynamic environments," *Machine Learning with Applications*, vol. 10, 2022, doi: 10.1016/j.mlwa.2022.100416.

[49] C. S. Weerasekera, Y. Latif, R. Garg and I. Reid, "Dense monocular reconstruction using surface normals," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2524-2531, 2017, doi: 10.1109/ICRA.2017.7989293.

[50] F. Liu, Chunhua Shen and Guosheng Lin, "Deep convolutional neural fields for depth estimation from a single image," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5162-5170, 2015, doi: 10.1109/CVPR.2015.7299152.

[51] D. Zoran, P. Isola, D. Krishnan and W. T. Freeman, "Learning Ordinal Relationships for Mid-Level Vision," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 388-396, 2015, doi: 10.1109/ICCV.2015.52.

[52] Peng Wang, Xiaohui Shen, Zhe Lin, S. Cohen, B. Price and A. Yuille, "Towards unified depth and semantic prediction from a single image," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 2800-2809, 2015, doi: 10.1109/CVPR.2015.7298897.

[53] D. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2650-2658, 2015, doi: 10.1109/ICCV.2015.304.

[54] F. Liu, C. Shen, G. Lin and I. Reid, "Learning Depth from Single Monocular Images Using Deep Convolutional Neural Fields," in *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 38, no. 10, pp. 2024-2039, 2016, doi: 10.1109/TPAMI.2015.2505283.

[55] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari and N. Navab, "Deeper Depth Prediction with Fully Convolutional Residual Networks," *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 239-248, 2016, doi: 10.1109/3DV.2016.32.

[56] F. Ma and S. Karaman, "Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image," *2018 IEEE International Conference on Robotics and Automation (ICRA),* pp. 4796-4803, 2018, doi: 10.1109/ICRA.2018.8460184.

[57] Z. Chen, V. Badrinarayanan, G. Drozdov, and A. Rabinovich, "Estimating Depth from RGB and Sparse Sensing," *Lecture Notes in Computer*

Science, vol. 11208, pp. 176–192, 2018, doi: 10.1007/978-3-030-01225-0_11.

[58] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia, "Unsupervised Learning of Geometry from Videos with Edge-Aware Depth-Normal Consistency," in *The Thirty-Second AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, pp. 7493–7500, 2018, doi: 10.1609/aaai.v32i1.12257.

[59] R. Mahjourian, M. Wicke and A. Angelova, "Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5667-5675, 2018, doi: 10.1109/CVPR.2018.00594.

[60] Z. Yin and J. Shi, "GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1983-1992, 2018, doi: 10.1109/CVPR.2018.00212.

[61] Y. Zou, Z. Luo, and J. B. Huang, "DF-Net: Unsupervised Joint Learning of Depth and Flow Using Cross-Task Consistency," *Lecture Notes in Computer Science*, vol. 11209, pp. 38–55, 2018, doi: 10.1007/978-3-030-01228-1_3.

[62] A. Ranjan *et al*., "Competitive Collaboration: Joint Unsupervised Learning of Depth, Camera Motion, Optical Flow and Motion Segmentation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 12232-12241, 2019, doi: 10.1109/CVPR.2019.01252.

[63] C. Godard, O. M. Aodha, M. Firman and G. Brostow, "Digging Into Self-Supervised Monocular Depth Estimation," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3827-3837, 2019, doi: 10.1109/ICCV.2019.00393.

[64] V. Guizilini, R. Ambruș, S. Pillai, A. Raventos and A. Gaidon, "3D Packing for Self-Supervised Monocular Depth Estimation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 2482-2491, 2020, doi: 10.1109/CVPR42600.2020.00256.

[65] C. Luo *et al*., "Every Pixel Counts ++: Joint Learning of Geometry and Motion with 3D Holistic Understanding," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 10, pp. 2624-2641, 2020, doi: 10.1109/TPAMI.2019.2930258.

[66] H. Y. Lee, H. W. Ho, and Y. Zhou, "Deep Learning-based Monocular Obstacle Avoidance for Unmanned Aerial Vehicle Navigation in Tree Plantations: Faster Region-based Convolutional Neural Network Approach," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 101, no. 5, 2021, doi: 10.1007/s10846-020-01284-z.

[67] A. Dosovitskiy *et al.,* "FlowNet: Learning Optical Flow with Convolutional Networks," *2015 IEEE International Conference on Computer Vision (ICCV),* pp. 2758-2766, 2015, doi: 10.1109/ICCV.2015.316.

[68] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy and T. Brox, "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1647-1655, 2017, doi: 10.1109/CVPR.2017.179.

[69] A. Ranjan and M. J. Black, "Optical Flow Estimation Using a Spatial Pyramid Network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 2720-2729, 2017, doi: 10.1109/CVPR.2017.291.

[70] D. Sun, X. Yang, M. -Y. Liu and J. Kautz, "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition,* pp. 8934-8943, 2018, doi: 10.1109/CVPR.2018.00931.

[71] Z. Teed and J. Deng, "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow," *Computer Vision – ECCV 2020*, vol. 12347, pp. 4839–4843, 2020, doi: 10.1007/978-3-030-58536-5_24.

[72] Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha, "Unsupervised deep learning for optical flow estimation," in *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, vol. 31, no. 1, pp. 1495–1501, 2017, doi: 10.1609/aaai.v31i1.10723.

[73] Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann, "Guided Optical Flow Learning," *arXiv*, 2017, doi: 10.48550/arXiv.1702.02295.

[74] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang and W. Xu, "Occlusion Aware Unsupervised Learning of Optical Flow," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition,* pp. 4884-4893, 2018, doi: 10.1109/CVPR.2018.00513.

[75] J. Janai, F. Güney, A. Ranjan, M. Black, and A. Geiger, "Unsupervised Learning of Multi-Frame Optical Flow with Occlusions," *Lecture Notes in Computer Science*, vol. 11220, pp. 713–731, 2018, doi: 10.1007/978-3-030-01270-0_42.

[76] Y. Zhong, P. Ji, J. Wang, Y. Dai and H. Li, "Unsupervised Deep Epipolar Flow for Stationary or Dynamic Scenes," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 12087-12096, 2019, doi: 10.1109/CVPR.2019.01237.

[77] B. Liao, J. Hu, and R. O. Gilmore, "Optical flow estimation combining with illumination adjustment and edge refinement in livestock UAV videos," *Computers and Electronics in Agriculture*, vol. 180, 2021, doi: 10.1016/j.compag.2020.105910.

[78] W. Yan, A. Sharma and R. T. Tan, "Optical Flow in Dense Foggy Scenes Using Semi-Supervised Learning," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 13256-13265, 2020, doi: 10.1109/CVPR42600.2020.01327.

[79] Q. Dai, V. Patii, S. Hecker, D. Dai, L. Van Gool and K. Schindler, "Self-supervised Object Motion and Depth Estimation from Video," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW),* pp. 4326-4334, 2020, doi: 10.1109/CVPRW50498.2020.00510.

[80] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A Naturalistic Open Source Movie for Optical Flow Evaluation," in *Computer Vision – ECCV 2012*, vol. 7577, pp. 611–625, 2012, doi: 10.1007/978-3-642-33783-3_44.

[81] S. Baker, S. Roth, D. Scharstein, M. J. Black, J. P. Lewis and R. Szeliski, "A Database and Evaluation Methodology for Optical Flow," *2007 IEEE 11th International Conference on Computer Vision*, pp. 1-8, 2007, doi: 10.1109/ICCV.2007.4408903.

[82] D. Berman, T. Treibitz and S. Avidan, "Single Image Dehazing Using Haze-Lines," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 3, pp. 720-734, 2020, doi: 10.1109/TPAMI.2018.2882478.

[83] C. Bailer, B. Taetz and D. Stricker, "Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4015-4023, 2015, doi: 10.1109/ICCV.2015.457.

[84] W. Hartmann, M. Havlena and K. Schindler, "Predicting Matchability," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9-16, 2014, doi: 10.1109/CVPR.2014.9.

[85] Y. Verdie, Kwang Moo Yi, P. Fua and V. Lepetit, "TILDE: A Temporally Invariant Learned DEtector," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5279-5288, 2015, doi: 10.1109/CVPR.2015.7299165.

[86] X. Shen *et al.,* "RF-Net: An End-To-End Image Matching Network Based on Receptive Field," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8124-8132, 2019, doi: 10.1109/CVPR.2019.00832.

[87] N. Jacobs, N. Roman and R. Pless, "Consistent Temporal Variations in Many Outdoor Scenes," *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-6, 2007, doi: 10.1109/CVPR.2007.383258.

[88] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, pp. 43–72, 2005, doi: 10.1007/s11263-005-3848-x.

[89] C. L. Zitnick and K. Ramnath, "Edge foci interest points," *2011 International Conference on Computer Vision*, pp. 359-366, 2011, doi: 10.1109/ICCV.2011.6126263.

[90] V. Balntas, K. Lenc, A. Vedaldi and K. Mikolajczyk, "HPatches: A Benchmark and Evaluation of Handcrafted and Learned Local Descriptors," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 3852-3861, 2017, doi: 10.1109/CVPR.2017.410.

[91] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," *Computer Vision – ECCV 2006*, vol. 3951, pp. 430–443, 2006, doi: 10.1007/11744023_34.

[92] W. Förstner, T. Dickscheid and F. Schindler, "Detecting interpretable and accurate scale-invariant keypoints," *2009 IEEE 12th International Conference on Computer Vision*, pp. 2256-2263, 2009, doi: 10.1109/ICCV.2009.5459458.

[93] P. Mainali, G. Lafruit, Q. Yang, B. Geelen, L. V. Gool, and R. Lauwereins, "SIFER: Scale-invariant feature detector with error resilience,"

International Journal of Computer Vision, vol. 104, no. 2, pp. 172–197, 2013, doi: 10.1007/s11263-013-0622-3.

[94] D. G. Low, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, 91–110, 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[95] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Computer Vision – ECCV 2006*, vol. 3951, pp. 404–417, 2006, doi: 10.1007/11744023_32.

[96] S. Salti, A. Lanza and L. Di Stefano, "Keypoints from symmetries by wave propagation," *2013 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 2898-2905, 2013, doi: 10.1109/CVPR.2013.373.

[97] C. L. Zitnick and K. Ramnath, "Edge foci interest points," *2011 International Conference on Computer Vision*, pp. 359-366, 2011, doi: 10.1109/ICCV.2011.6126263.

[98] Y. Tian, B. Fan and F. Wu, "L2-Net: Deep Learning of Discriminative Patch Descriptor in Euclidean Space," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6128-6136, 2017, doi: 10.1109/CVPR.2017.649.

[99] A. Mishchuk, D. Mishkin, F. Radenović, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," *Advances in neural information processing*, vol. 30, pp. 1–12, 2017.

[100] Y. Ono, P. Fua, E. Trulls, and K. M. Yi, "LF-Net: Learning local features from images," *Advances in Neural Information Processing Systems*, vol. 31, pp. 6234–6244, 2018.

[101] Z. Qin, M. Yin, G. Li, and F. Y. A, "SP-Flow: Self-supervised optical flow correspondence point prediction for real-time SLAM," *Computer Aided Geometric Design*, vol. 82, 2020, doi: 10.1016/j.cagd.2020.101928.

[102] R. Mur-Artal and J. D. Tardós, "Visual-Inertial Monocular SLAM With Map Reuse," in *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796-803, 2017, doi: 10.1109/LRA.2017.2653359.

[103] H. M. S. Bruno and E. L. Colombini, "LIFT-SLAM: A deep-learning feature-based monocular visual SLAM method," *Neurocomputing*, vol. 455, pp. 97–110, 2021, doi: 10.1016/j.neucom.2021.05.027.

[104] R. Mur-Artal, J. M. M. Montiel and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," in *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015, doi: 10.1109/TRO.2015.2463671.

[105] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, 2016, doi: 10.1177/0278364915620033.

[106] J. Tang, L. Ericson, J. Folkesson and P. Jensfelt, "GCNv2: Efficient Correspondence Prediction for Real-Time SLAM," in *IEEE Robotics and Automation Letters,* vol. 4, no. 4, pp. 3505-3512, 2019, doi: 10.1109/LRA.2019.2927954.

[107] Y. Sun, M. Liu, and M. Q. Meng, "Improving RGB-D SLAM in dynamic environments : A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110–122, 2017, doi: 10.1016/j.robot.2016.11.012.

[108] M. Kaneko, K. Iwami, T. Ogawa, T. Yamasaki and K. Aizawa, "Mask-SLAM: Robust Feature-Based Monocular SLAM by Masking Using Semantic Segmentation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW),* pp. 371-3718, 2018, doi: 10.1109/CVPRW.2018.00063.

[109] L. -C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834-848, 2018, doi: 10.1109/TPAMI.2017.2699184.

[110] C. Yu *et al*., "DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1168-1174, 2018, doi: 10.1109/IROS.2018.8593691.

[111] B. Bescos, J. M. Fácil, J. Civera and J. Neira, "DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes," in *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076-4083, 2018, doi: 10.1109/LRA.2018.2860039.

[112] F. Zhong, S. Wang, Z. Zhang, C. Chen and Y. Wang, "Detect-SLAM: Making Object Detection and SLAM Mutually Beneficial," *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1001-1010, 2018, doi: 10.1109/WACV.2018.00115.

[113] G. Tian, L. Liu, J. H. Ri, Y. Liu, and Y. Sun, "ObjectFusion: An object detection and segmentation framework with RGB-D SLAM and convolutional neural networks," *Neurocomputing*, vol. 345, pp. 3–14, 2019, doi: 10.1016/j.neucom.2019.01.088.

[114] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Improving monocular visual SLAM in dynamic environments: An optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019, doi: 10.1080/01691864.2019.1610060.

[115] C. Shao, C. Zhang, Z. Fang and G. Yang, "A Deep Learning-Based Semantic Filter for RANSAC-Based Fundamental Matrix Calculation and the ORB-SLAM System," in *IEEE Access*, vol. 8, pp. 3212-3223, 2020, doi: 10.1109/ACCESS.2019.2962268.

[116] L. Xua, C. Feng, V. R. Kamata, and C. C. Menassaa, "A scene-adaptive descriptor for visual SLAM-based locating applications in built environments," *Automation in Construction*, vol. 112, 2020, doi: 10.1016/j.autcon.2019.103067.

[117] W. Liu, Y. Mo, J. Jiao and Z. Deng, "EF-Razor: An Effective Edge-Feature Processing Method in Visual SLAM," in *IEEE Access*, vol. 8, pp. 140798-140805, 2020, doi: 10.1109/ACCESS.2020.3013806.

[118] I. Rusli, B. R. Trilaksono and W. Adiprawita, "RoomSLAM: Simultaneous Localization and Mapping With Objects and Indoor Layout Structure," in *IEEE Access*, vol. 8, pp. 196992-197004, 2020, doi: 10.1109/ACCESS.2020.3034537.

[119] S. J. A, L. Chen, R. S. A, and S. McLoone, "A novel vSLAM framework with unsupervised semantic segmentation based on adversarial transfer learning," *Applied Soft Computing Journal*, vol. 90, 2020, doi: 10.1016/j.asoc.2020.106153.

[120] X. Zhao, C. Wang and M. H. Ang, "Real-Time Visual-Inertial Localization Using Semantic Segmentation Towards Dynamic Environments," in *IEEE Access*, vol. 8, pp. 155047-155059, 2020, doi: 10.1109/ACCESS.2020.3018557.

[121] T. Qin, P. Li and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," in *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004-1020, 2018, doi: 10.1109/TRO.2018.2853729.

[122] J. Cheng, Z. Wang, H. Zhou, L. L. 1, and J. Yao, "DM-SLAM : A Feature-Based SLAM System for Rigid Dynamic Scenes," *International Journal of Geo-Information*, vol. 9, no. 4, pp. 1–18, 2020, doi: 10.3390/ijgi9040202.

[123] Y. Liu and J. Miura, "RDS-SLAM: Real-Time Dynamic SLAM Using Semantic Segmentation Methods," in *IEEE Access*, vol. 9, pp. 23772-23785, 2021, doi: 10.1109/ACCESS.2021.3050617.

[124] P. Su, S. Luo and X. Huang, "Real-Time Dynamic SLAM Algorithm Based on Deep Learning," in *IEEE Access*, vol. 10, pp. 87754-87766, 2022, doi: 10.1109/ACCESS.2022.3199350.

[125] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," *arXiv*, pp. 1–16, 2017.

[126] A. Handa, T. Whelan, J. McDonald and A. J. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1524-1531, 2014, doi: 10.1109/ICRA.2014.6907054.

[127] M. Grupp, *evo: Python package for the evaluation of odometry and slam*, 2017, https://github.com/MichaelGrupp/evo.

[128] Z. X. Zou, S. S. Huang, T. J. Mu, and Y. P. Wang, "ObjectFusion: Accurate object-level SLAM with neural object priors," *Graphical Models*, vol. 123, 2022, doi: 10.1016/j.gmod.2022.101165.

[129] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison and S. Leutenegger, "MID-Fusion: Octree-based Object-Level Multi-Instance Dynamic SLAM," *2019 International Conference on Robotics and Automation (ICRA),* pp. 5231-5237, 2019, doi: 10.1109/ICRA.2019.8794371.

[130] Y. Zhu, R. Gao, S. Huang, S. -C. Zhu and Y. N. Wu, "Learning Neural Representation of Camera Pose with Matrix Representation of Pose Shift via View Synthesis," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9954-9963, 2021, doi: 10.1109/CVPR46437.2021.00983.

[131] C. Qiao, Z. Xiang, and X. Wang, "Objects Matter: Learning Object Relation Graph for Robust Camera Relocalization," *arXiv*, 2022.

[132] J. McCormac, A. Handa, S. Leutenegger and A. J. Davison, "SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-

training on Indoor Segmentation?," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2697-2706, 2017, doi: 10.1109/ICCV.2017.292.

[133] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi and A. Fitzgibbon, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930-2937, 2013, doi: 10.1109/CVPR.2013.377.

[134] M. Runz, M. Buffier and L. Agapito, "MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects," *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 10-20, 2018, doi: 10.1109/ISMAR.2018.00024.

[135] O. Kähler, V. Adrian Prisacariu, C. Yuheng Ren, X. Sun, P. Torr and D. Murray, "Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 11, pp. 1241-1250, 2015, doi: 10.1109/TVCG.2015.2459891.

[136] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration," *arXiv*, pp. 36–47, 2017, doi: 10.5244/C.31.158.

[137] A. Kendall and R. Cipolla, "Geometric Loss Functions for Camera Pose Regression with Deep Learning," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6555-6564, 2017, doi: 10.1109/CVPR.2017.694.

[138] S. Brahmbhatt, J. Gu, K. Kim, J. Hays and J. Kautz, "Geometry-Aware Learning of Maps for Camera Localization," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2616-2625, doi: 10.1109/CVPR.2018.00277.

[139] C. Zhao, L. Sun, Z. Yan, G. Neumann, T. Duckett, and R. Stolkin, "Learning Kalman Network: A deep monocular visual odometry for on-road driving," *Robotics and Autonomous Systems*, vol. 121, 2019, doi: 10.1016/j.robot.2019.07.004.

[140] C. Tao, Z. Gao, J. Yan, C. Li and G. Cui, "Indoor 3D Semantic Robot VSLAM Based on Mask Regional Convolutional Neural Network," in *IEEE Access*, vol. 8, pp. 52906-52916, 2020, doi: 10.1109/ACCESS.2020.2981648.

[141] X. Han, S. Li, X. Wang, and W. Zhou, "Semantic mapping for mobile robots in indoor scenes: A survey," *Information*, vol. 12, no. 2, pp. 1–14, 2021, doi: 10.3390/info12020092.

[142] J. McCormac, A. Handa, A. Davison and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4628-4635, 2017, doi: 10.1109/ICRA.2017.7989538.

[143] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford and I. Reid, "Meaningful maps with object-oriented semantic mapping," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5079-5085, 2017, doi: 10.1109/IROS.2017.8206392.

[144] S. Yang, Y. Huang and S. Scherer, "Semantic 3D occupancy mapping through efficient high order CRFs," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 590-597, 2017, doi: 10.1109/IROS.2017.8202212.

[145] M. Grinvald *et al.*, "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery," in *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037-3044, 2019, doi: 10.1109/LRA.2019.2923960.

[146] P. Karkus, A. Angelova, V. Vanhoucke and R. Jonschkowski, "Differentiable Mapping Networks: Learning Structured Map Representations for Sparse Visual Localization," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4753-4759, 2020, doi: 10.1109/ICRA40945.2020.9197452.

[147] A. Geiger, J. Ziegler and C. Stiller, "StereoScan: Dense 3d reconstruction in real-time," *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 963-968, 2011, doi: 10.1109/IVS.2011.5940405.

[148] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel, "Backprop KF: Learning discriminative deterministic state estimators," *Advances in Neural Information Processing Systems*, vol. 29, pp. 4383–4391, 2016.

[149] H. Coskun, F. Achilles, R. DiPietro, N. Navab and F. Tombari, "Long Short-Term Memory Kalman Filters: Recurrent Neural Estimators for Pose Regularization," *2017 IEEE International Conference on Computer Vision (ICCV),* pp. 5525-5533, 2017, doi: 10.1109/ICCV.2017.589.

[150] Y. Hou, H. Zhang and S. Zhou, "Convolutional neural network-based image representation for visual loop closure detection," *2015 IEEE International Conference on Information and Automation*, pp. 2238-2245, 2015, doi: 10.1109/ICInfA.2015.7279659.

[151] Y. Xia, J. Li, L. Qi, H. Yu and J. Dong, "An Evaluation of Deep Learning in Loop Closure Detection for Visual SLAM," *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 85-91, 2017, doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.18.

[152] X. Zhang, Y. Su and X. Zhu, "Loop closure detection for visual SLAM systems using convolutional neural network," *2017 23rd International Conference on Automation and Computing (ICAC)*, pp. 1-6, 2017, doi: 10.23919/IConAC.2017.8082072.

[153] N. Merrill and G. Huang, "Lightweight Unsupervised Deep Loop Closure," *Robotics: Science and Systems*, 2018.

[154] A. R. Memon, H. Wang, and A. Hussain, "Loop closure detection using supervised and unsupervised deep neural networks for monocular SLAM systems," *Robotics and Autonomous Systems*, vol. 126, 2020, doi: 10.1016/j.robot.2020.103470.

[155] J. Chang, N. Dong, D. Li, and M. Qin, "Triplet loss based metric learning for closed loop detection in VSLAM system," *Expert Systems with Applications*, vol. 185, 2021, doi: 10.1016/j.eswa.2021.115646.

[156] R. Duan, Y. Feng, and C.-Y. Wen, "Deep Pose Graph-Matching-Based Loop Closure Detection for Semantic Visual SLAM," *Sustainability*, vol. 14, no. 9, 2022, doi: 10.3390/su141911864.

[157] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, 2008, doi: 10.1177/0278364908090961.

[158] D. Galvez-López and J. D. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," in *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188-1197, 2012, doi: 10.1109/TRO.2012.2197158.

[159] rmsalinas, *DBoW3 dbow3*, 2017, https://github.com/rmsalinas/DBow3.

[160] E. Garcia-Fidalgo and A. Ortiz, "iBoW-LCD: An Appearance-Based Loop-Closure Detection Approach Using Incremental Bags of Binary Words," in *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051-3057, 2018, doi: 10.1109/LRA.2018.2849609.

[161] P. -E. Sarlin, C. Cadena, R. Siegwart and M. Dymczyk, "From Coarse to Fine: Robust Hierarchical Localization at Large Scale," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12708-12717, 2019, doi: 10.1109/CVPR.2019.01300.

[162] A. Kendall, M. Grimes and R. Cipolla, "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2938-2946, 2015, doi: 10.1109/ICCV.2015.336.

[163] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4762-4769, 2016, doi: 10.1109/ICRA.2016.7487679.

[164] I. Melekhov, J. Ylioinas, J. Kannala and E. Rahtu, "Image-Based Localization Using Hourglass Networks," *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 870-877, 2017, doi: 10.1109/ICCVW.2017.107.

[165] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck and D. Cremers, "Image-Based Localization Using LSTMs for Structured Feature Correlation," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 627-637, 2017, doi: 10.1109/ICCV.2017.75.

[166] J. Wu, L. Ma and X. Hu, "Delving deeper into convolutional neural networks for camera relocalization," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5644-5651, 2017, doi: 10.1109/ICRA.2017.7989663.

[167] X. Wang, X. Wang, C. Wang, X. Bai, and J. Wu, "Discriminative Features Matter : Multi-layer Bilinear Pooling for Camera Localization," in *In: British Machine Vision Conference*, pp. 1–12, 2019.

[168] M. Bui, C. Baur, N. Navab, S. Ilic and S. Albarqouni, "Adversarial Networks for Camera Pose Regression and Refinement," *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 3778-3787, 2019, doi: 10.1109/ICCVW.2019.00470.

[169] M. Cai, C. Shen, and I. Reid, "A hybrid probabilistic model for camera relocalization," in *British Machine Vision Conference 2018, BMVC 2018*, pp. 1–12, 2019.

[170] S. Saha, G. Varma, and C. V. Jawahar, "Improved visual relocalization by discovering anchor points," in *British Machine Vision Conference 2018, BMVC 2018*, pp. 1–11, 2018.

[171] B. Wang, C. Chen, C. X. Lu, P. Zhao, N. Trigoni, and A. Markham, "AtLoc: Attention guided camera localization," in *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, vol. 34, no. 6, 2020, doi: 10.1609/aaai.v34i06.6608.

[172] M. O. Turkoglu, E. Brachmann, K. Schindler, G. J. Brostow and A. Monszpart, "Visual Camera Re-Localization Using Graph Neural Networks and Relative Pose Supervision," *2021 International Conference on 3D Vision (3DV)*, pp. 145-155, 2021, doi: 10.1109/3DV53792.2021.00025.

[173] J. Engel, V. Koltun and D. Cremers, "Direct Sparse Odometry," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611-625, 2018, doi: 10.1109/TPAMI.2017.2658577.

[174] N. Fanani, A. Stürck, M. Ochs, H. Bradler, and R. Mester, "Predictive monocular odometry (PMO): What is possible without RANSAC and multiframe bundle adjustment?," *Image and Vision Computing*, vol. 68, pp. 3–13, 2017, doi: 10.1016/j.imavis.2017.08.002.

[175] R. Wang, M. Schwörer and D. Cremers, "Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3923-3931, 2017, doi: 10.1109/ICCV.2017.421.

[176] F. Xue, Q. Wang, X. Wang, W. Dong, J. Wang, and H. Zha, "Guided Feature Selection for Deep Visual Odometry," *Lecture Notes in Computer Science*, vol. 11366, pp. 293–308, 2019, doi: 10.1007/978-3-030-20876-9_19.

[177] R. Kreuzig, M. Ochs and R. Mester, "DistanceNet: Estimating Traveled Distance From Monocular Images Using a Recurrent Convolutional Neural Network," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1258-1266, 2019, doi: 10.1109/CVPRW.2019.00165.

[178] A. Saxena, S. H. Chung, and A. Y. Ng, "3-D Depth Reconstruction from a Single Still Image," *International Journal of Computer Vision*, vol. 76, pp. 53–69, 2008, doi: 10.1007/s11263-007-0071-y.

[179] A. Roy and S. Todorovic, "Monocular Depth Estimation Using Neural Regression Forest," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5506-5514, 2016, doi: 10.1109/CVPR.2016.594.

[180] M. Mancini, G. Costante, P. Valigi and T. A. Ciarfuglia, "Fast robust monocular depth estimation for Obstacle Detection with fully convolutional networks," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4296-4303, 2016, doi: 10.1109/IROS.2016.7759632.

[181] C. Cadena, A. Dick, and I. D. Reid, "Multi-modal auto-encoders as joint estimators for robotics scene understanding," *Robotics: Science and Systems*, vol. 12, 2016, doi: 10.15607/rss.2016.xii.041.

[182] D. Xu, E. Ricci, W. Ouyang, X. Wang and N. Sebe, "Multi-scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 161-169, 2017, doi: 10.1109/CVPR.2017.25.

[183] Y. Liao, L. Huang, Y. Wang, S. Kodagoda, Y. Yu and Y. Liu, "Parse geometry from a line: Monocular depth estimation with partial laser observation," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5059-5066, 2017, doi: 10.1109/ICRA.2017.7989590.

[184] D. Xu, W. Wang, H. Tang, H. Liu, N. Sebe and E. Ricci, "Structured Attention Guided Convolutional Neural Fields for Monocular Depth Estimation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3917-3925, 2018, doi: 10.1109/CVPR.2018.00412.

[185] Y. Li, K. Qian, T. Huang, and J. Zhou, "Depth estimation from monocular image and coarse depth points based on conditional GAN," in *MATEC Web of Conferences*, vol. 175, pp. 1–5, 2018, doi: 10.1051/matecconf/201817503055.

[186] A. Wang, Z. Fang, Y. Gao, X. Jiang and S. Ma, "Depth Estimation of Video Sequences With Perceptual Losses," in *IEEE Access*, vol. 6, pp. 30536-30546, 2018, doi: 10.1109/ACCESS.2018.2846546.

[187] D. Wofk, F. Ma, T. -J. Yang, S. Karaman and V. Sze, "FastDepth: Fast Monocular Depth Estimation on Embedded Systems," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6101-6108, 2019, doi: 10.1109/ICRA.2019.8794182.

[188] S. Gur and L. Wolf, "Single Image Depth Estimation Trained via Depth From Defocus Cues," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7675-7684, 2019, doi: 10.1109/CVPR.2019.00787.

[189] d. xu, E. Ricci, W. Ouyang, X. Wang and N. Sebe, "Monocular Depth Estimation Using Multi-Scale Continuous CRFs as Sequential Deep Networks," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 6, pp. 1426-1440, 2019, doi: 10.1109/TPAMI.2018.2839602.

[190] X. Tu, C. Xu, S. Liu, G. Xie and R. Li, "Real-Time Depth Estimation with an Optimized Encoder-Decoder Architecture on Embedded Devices," *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS),* pp. 2141-2149, 2019, doi: 10.1109/HPCC/SmartCity/DSS.2019.00296.

[191] T. -H. Wang, F. -E. Wang, J. -T. Lin, Y. -H. Tsai, W. -C. Chiu and M. Sun, "Plug-and-Play: Improve Depth Prediction via Sparse Data Propagation," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5880-5886, 2019, doi: 10.1109/ICRA.2019.8794404.

[192] J. Hu, Y. Zhang and T. Okatani, "Visualization of Convolutional Neural Networks for Monocular Depth Estimation," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3868-3877, 2019, doi: 10.1109/ICCV.2019.00397.

[193] X. Tu *et al*., "Learning Depth for Scene Reconstruction Using an Encoder-Decoder Model," in *IEEE Access*, vol. 8, pp. 89300-89317, 2020, doi: 10.1109/ACCESS.2020.2993494.

[194] M. Weber, C. Rist and J. M. Zöllner, "Learning temporal features with CNNs for monocular visual ego motion estimation," *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1-6, 2017, doi: 10.1109/ITSC.2017.8317922.

[195] S. Wang, R. Clark, H. Wen and N. Trigoni, "DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2043-2050, 2017, doi: 10.1109/ICRA.2017.7989236.

[196] V. Peretroukhin, L. Clement and J. Kelly, "Reducing drift in visual odometry by inferring sun direction using a Bayesian Convolutional Neural Network," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2035-2042, 2017, doi: 10.1109/ICRA.2017.7989235.

[197] R. Li, S. Wang, Z. Long and D. Gu, "UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7286-7291, 2018, doi: 10.1109/ICRA.2018.8461251.

[198] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal and I. M. Reid, "Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 340-349, 2018, doi: 10.1109/CVPR.2018.00043.

[199] E. J. Shamwell, S. Leung and W. D. Nothwang, "Vision-Aided Absolute Trajectory Estimation Using an Unsupervised Deep Network with Online Error Correction," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2524-2531, 2018, doi: 10.1109/IROS.2018.8593573.

[200] N. Yang, L. von Stumberg, R. Wang and D. Cremers, "D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1278-1289, 2020, doi: 10.1109/CVPR42600.2020.00136.

[201] Y. A. M. Turan, A. E. Sarı, M. R. U. Saputra, P. P. B. de Gusmo, A. Markham, and N. Trigoni, "SelfVIO: Self-Supervised Deep Monocular Visual-Inertial Odometry and Depth Estimation," *Neural Networks*, vol. 150, pp. 119–136, 2022, doi: 10.1016/j.neunet.2022.03.005.

[202] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, 2014, doi: 10.1177/0278364914554813.

[203] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "IEKF-based Visual-Inertial Odometry using Direct Photometric Feedback," *The*

International Journal of Robotics Research, vol. 36, no. 10, pp. 1053–1072, 2017, doi: 10.3929/ethz-b-000187364.

[204] Y. Xiao, L. Li, X. Li and J. Yao, "DeepMLE: A Robust Deep Maximum Likelihood Estimator for Two-view Structure from Motion," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10643-10650, 2022, doi: 10.1109/IROS47612.2022.9981975.

[205] G. Zhai, L. Liu, L. Zhang, Y. Liu, and Y. Jiang, "PoseConvGRU: A Monocular Approach for Visual Ego-motion Estimation by Learning," *Pattern Recognition*, vol. 102, 2020, doi: 10.1016/j.patcog.2019.107187.

[206] R. Zhu, M. Yang, W. Liu, R. Song, B. Yan, and Z. Xiao, "DeepAVO: Efficient pose refining with feature distilling for deep visual odometry," *Neurocomputing*, vol. 467, pp. 22–35, 2022, doi: 10.1016/j.neucom.2021.09.029.

[207] F. A. Muhammet, D. Akif, Y. Abdullah, and Y. Alper, "HVIOnet: A deep learning based hybrid visual–inertial odometry approach for unmanned aerial system position estimation," *Neural Networks*, vol. 155, pp. 461-474, 2022, doi: 10.1016/j.neunet.2022.09.001.

[208] X. Haixin, L. Yiyou, H. Zeng, Q. Li, H. Liu, B. Fan, and C. Li, "Robust self-supervised monocular visual odometry based on prediction-update pose estimation network," *Engineering Applications of Artificial Intelligence*, vol. 116, 2022, doi: 10.1016/j.engappai.2022.105481.

[209] Y. Lu, Y. Chen, D. Zhao, and D. Li, "MGRL: Graph neural network based inference in a Markov network with reinforcement learning for visual navigation," *Neurocomputing*, vol. 421, pp. 140–150, 2021, doi: 10.1016/j.neucom.2020.07.091.

[210] A. J. Ali, M. Kouroshli, S. Semenova, Z. S. Hashemifar, S. Y. Ko, and K. Dantu, "Edge-SLAM: Edge-Assisted Visual Simultaneous Localization and Mapping," *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 1, pp. 1–31, 2022, doi: 10.1145/3561972.

[211] M. Kegeleirs, G. Grisetti, and M. Birattari, "Swarm SLAM: Challenges and Perspectives," *Frontiers in Robotics and AI*, vol. 8, pp. 1–6, 2021, doi: 10.3389/frobt.2021.618268.

[212] P. -Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone and G. Beltrame, "DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams," in *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656-1663, 2020, doi: 10.1109/LRA.2020.2967681.

[213] M. N. Osman Zahid and L. J. Hao, "A Study on Obstacle Detection For IoT Based Automated Guided Vehicle (AGV)," *Mekatronika*, vol. 4, no. 1, pp. 30–41, 2022, doi: 10.15282/mekatronika.v4i1.7534.

[214] S. Buck, R. Hanten, K. Bohlmann and A. Zell, "Generic 3D obstacle detection for AGVs using time-of-flight cameras," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4119-4124, 2016, doi: 10.1109/IROS.2016.7759606.